Navn og kandidatnummer
Jonas Kalmar Rønning, 8031
Christian Nicolai Iversen, 8126
Kristoffer Kittilsen, 8134

Emnekode: BAO301
Emnenavn: Bachelorprosjekt
Oppdragsgiver: Watec Solutions AS
Innleveringsdato: 20.05.2021
Antall sider: 72
Antall ord: 11990

Tilgjengelighet: **Fri** ☑ Begrenset ☐

Høyskolen Kristiania

Fra e-post til web: Utviklingen av en B2B e-commerceplattform
From e-mail to web: Development of a B2B e-commerce platform

Våren 2021

Sammendrag

Denne rapporten omhandler vår prosess under utvikling av en e-commerceplattform for Watec Solutions sine bedriftskunder. Målet var å føre bestillingene de i dag håndterer over e-post til en webplattform for å spare tid, ha bedre oversikt over alle åpne bestillinger og forenkle bestillingsprosessen for deres kunder. Prosjektet har også bestått av en forskningsfase og brukertester, hvor vår hensikt var å samle data som kunne hjelpe oss med viktige valg underveis. Resultatet av prosjektet er en webapp i React som blir servert av en Node.js backend som kjører på Heroku.

1 Innledning

Denne oppgaven er skrevet som en del av faget BAO301 Bachelorprosjekt. I dette faget har vi jobbet i gruppe for å løse et problem ute i bedrift. Formålet med prosjektet er å få yrkeserfaring gjennom å jobbe ute i bedrift og samtidig vise at vi kan planlegge og gjennomføre et større prosjekt. Vi har vært utplassert hos Watec Solutions. Vi har fått i oppgave å lage en e-commerceplattform som har til hensikt å gjøre ordreprosessen enklere for oppdragsgiver og deres kunder. Watec Solutions lever varer og tjenester til et bredt spekter av helt forskjellige kunder. Dette har skapt mange interessante problemstillinger som vi har fått bryne oss på. Vårt mål vil være å utvikle en løsning som kan flytte ordreprosessen fra e-post som Watec bruker i dag til en mer brukervennlig online platform.

I denne rapporten vil vi presentere og begrunne våre valg underveis, samt vise frem den endelige løsningen og dens svakheter. Vi kommer også til å dele noen tanker rundt løsningens nytteverdi for bedriften og hvilke videreutviklingspotensiale den har.

1.1 Presentasjon av gruppen

Gruppen består av tre teknologistudenter fra Høyskolen Kristiania. Gruppen ble etablert allerede tidlig høsten 2020 fordi vi visste at hvert gruppemedlem satt med god teknisk kompetanse og interpersonlige ferdigheter. Vi mener at som gruppe utfyller vi hverandre godt og har lignende interesser, som ville gjøre det lettere å finne et prosjekt som alle ville finne engasjerende.

Navn	Linje	Rolle
Christian Nicolai Iversen	Frontend- og Mobilutvikling	Utvikler og Product Owner
Jonas Kalmar Rønning	Frontend- og Mobilutvikling	Scrum Master, Utvikler og Designer
Kristoffer Kittilsen	Frontend- og Mobilutvikling	Utvikler og Designer

Tabell 1 - Medlemmer på gruppen

Siden vi alle har mest erfaring fra web- og mobilapplikasjoner på grunn av studier ønsket vi å finne en oppgave hvor vi kunne utviklet dette. Vi har også to på gruppen som har tidligere erfaring fra salg og bruk av CRM-systemer, samt et medlem som har vært med å utvikle en bestillingsplattform i en startup. Derfor så vi helst for oss et prosjekt hvor vi kunne utvikle en type bedriftsapplikasjon som skulle kjøres på web eller i en mobilapp.

Vi var i kontakt med noen bedrifter som hadde prosjekter som vi trodde kunne være relevant, og endte opp hos Watec da prosjektet de ville utføre var helt innenfor det vi så etter, og det var god kommunikasjon mellom gruppen og selskapet.

1.2 Presentasjon av oppdragsgiver

Watec Solutions er et selskap i Lillestrøm opprettet i 2016 som leverer ulike tjenester og produkter for bedrifter. Selskapets overordnet mål er å levere høyteknologiske produkter og tjenester som er kostbesparende for selskaper og miljøvennlig for samfunnet. Watec prioriterer gode kundeforhold og kundeopplevelser; selskapet har størst fortjeneste på volumsalg hos gjentagende kunder. I de siste årene har mesteparten av arbeidet til Watec gått ut på glasslipingstjenester i anleggsbransjen. Dette går ut på å slipe vekk skraper og skader i glassplater i stedet for å bytte ut hele glasset, som er både billigere for entreprenøren samt et mer miljøvennlig alternativ. I tillegg til dette driver selskapet med forskning og utvikling av andre løsninger, og har ambisjoner om å jobbe inn mot flere bransjer. Blant annet jobber selskapet med å utvikle kjemi for å forhindre korrosjon av metaller, samt produksjon av maskiner som gjenvinner plast til olje.

1.3 Presentasjon av prosjektet

Watec Solutions ønsker å gjøre sin ordreprosess enklere ved å ta i bruk et ordrestyringssystem. Per i dag skjer all kommunikasjon med kunder over epost, og dette er veldig uoversiktlig både for kunder og for bedriften. Vi har fått i oppgave å utvikle et system skreddersydd for Watec og deres kunder. Systemet vi lager vil være tilgjengelig for kundene gjennom en nettside hvor inviterte brukere kan logge seg på og legge inn ordre.

På denne nettsiden vil det være mulig for oppdragsgiver å se alle ordre. De skal også kunne se en oversikt over kunder og hvem kunder som har bestilt hva. Siden Watec utelukkende driver med B2B-salg skal systemet være lukket. Alle kunder må inviteres av oppdragsgiver før de kan registrere en bruker. Kundene vil etter å ha registrert seg få tilgang til ordresystemet hvor de kan bestille varer.
Løsningen må være sikker og må også legge opp til at bestillinger kan digitalt signeres med tredjeparts tjenester som f.eks DocuSign. Den skal også føre statistikk over ordre, samt tilby kundestatistikk. Vi skal også hjelpe bedriften med å sette opp løsningen på en skyserver.

1.4 Målsetninger

Et av målene vi som en gruppe har er å lage en løsning som Watec Solutions kan ha nytte av og vil gjøre det lettere for bedriften å holde styr på ordre ved bruk av et nytt ordrestyringssystem. Gruppen vil også at løsningen fyller alle tekniske krav, samt at det er av høy kvalitet.

Løsningen vi lager skal utvikles på en slik måte at vi kan skrive en god bacheloroppgave rundt denne løsningen. Gjennom prosjektet ønsker vi som gruppe å lære mer om hvordan det er å jobbe i slike prosjekter i arbeidslivet og lære hvordan det er å jobbe med andre i større prosjekter.

1.5 Problemstilling

Oppdragsgiver har vært klar over at det er forbedringspotensiale i ordreprosessen de har i dag. De har gitt uttrykk for at de ønsker at vi ser på prosessen de gjør i dag, finner ut hva som tar mest tid, og hvordan ting kan forenkles via en skreddersydd digital plattform. Resultatet av prosjektet vil være en 'betaversjon' av en digital salgsplattform som viser hvordan vi har løst problemene de har med bestillinger i dag, og som de kan teste med noen kunder allerede når prosjektet er fullført. Vi ønsker også å forsikre oss om at vi kan utvikle et brukervennlig system som ikke går negativt utover kundeopplevelsen, og som kan gjøre det lettere for kunden å bestille og holde oversikt over tidligere bestillinger. Alt i alt vil dette sørge for at bedriften kan fokusere på å utvikle og levere gode produkter og god service uten å måtte bruke unødvendig mye tid på oppfølging av e-poster. På bakgrunn av dette har vi utarbeidet en problemstilling:

Hvordan kan vi utvikle en smartere løsning for å håndtere ordre som sparer bedriften for tid og penger, uten å gi negativt utslag på de gode kundeforholdene de har i dag?

2 Forankring i forskningsområde

I dette kapittelet skriver vi om relevant teori rundt begrepene som blir brukt og som ligger til grunnlag for den tekniske løsningen.

2.1 Design og utforming

For å utvikle en løsning som oppfyller kravene til vår oppdragsgiver trenger vi å tilegne oss kunnskap innenfor flere felt. Vi har særlig hatt fokus på brukervennlighet og vi har valgt å lene oss på boken «The Design of Everyday Things» av designforskeren Don Norman. I boken redegjør han for seks designprinsipper vi har lent oss på når det kommer til planlegging og design av grensesnittene i prosjektet vårt (Norman 2013, 10).

Synlighet
Designprinsippet synlighet, eller visibility, handler om at brukeren ser hvilket funksjoner som er tilgjengelig til enhver tid. I konteksten av en webapplikasjon vil det for eksempel være viktig at ikoner blir brukt til å representere noe som er intuitivt for nesten alle brukere.

Hint
Designprinsippet hint, eller affordance, handler om at designet på en ting skal være selvforklarende og gi hint om hvordan tingen bør brukes. Det mest kjente eksempelet på hva som skjer når man ikke har hint er ‘Norman Doors’, et begrep kalt opp etter Don Norman, som beskriver svingdører som ikke gjør det tydelig hvilken side som skal dyttes eller dras på, gjerne ved å ha håndtak på begge sider når det kun er behov på siden man skal dra.

Begrensninger
Designprinsippet begrensninger, eller constraints, handler om å begrense mulighetene til en bruker for å forminske risikoen for at brukeren gjør feil som hindrer at de kommer seg videre.

Tilbakemelding

Designprinsippet tilbakemelding, eller feedback, handler om å gi brukeren tilbakemelding på hva de har gjort og hvilken effekt det har hatt. I kontekst av en applikasjon er det viktig å gi brukeren tilbakemelding hver gang de har gjort en handling som har endret tilstanden til appen eller databasen, for eksempel ved å ha opprettet en ordre, gjerne med noe forklarende tekst på hva som har skjedd.

Sammenheng

Designprinsippet sammenheng, eller mapping, handler om sammenhengen mellom kontroll og effekt. Ideen er at med god design skal kontrollene representere det de kontrollerer.

Konsistens

Designprinsippet konsistens, eller consistency, handler om at det skal være et lignende design på ting som har lignende funksjonalitet. For eksempel er det lurt at knapper for å opprette nye ting i applikasjonen ligner på hverandre, men det bør være forskjell på knappene som oppretter ny data og som slettere eksisterende data. Vi kan også skille mellom **intern** og **ekstern** konsistens. Ekstern konsistens handler om å tilnærme seg konvensjoner vi har på ting, for eksempel bør man aldri bruke en rød knapp til å representere ‘ja’ mens en grønn knapp til å representere ‘nei’, da disse fargene tolkes på motsatt måte av allmennheten. Intern konsistens handler derimot om å forsikre at utformingen er tilnærmet lik under hele applikasjonen, for eksempel ved at feilmeldinger alltid vises på samme måte og samme sted.

2.2 B2B ERP

Vi har også sett på hvilken verdi den endelige løsningen kan ha for bedriften ved å kikke på tidligere forskning. I en artikkel skrevet av Kenneth E. Murphy og Steven John Simon ble det undersøkt hvordan man kan måle verdien i et ERP-system. Her kom de fram til at en 5% forbedring av kundeopplevelsen førte til en 1% økning av markedsandeler for deres casebedrift. Vår løsning vil ikke bli et ERP-system, men vi synes det er verdt å ta med siden vi lager en salgsplattform. I undersøkelsen

understreker de at de største forbedringene kom fra markedsføring og salg (Murphy og Simon 2010, 8)

Viktigheten av kundeopplevelse er også understreket i andre forskningsartikler vi har sett på. En undersøkelse av Graeme McLean i 2017 hevder at kvaliteten på informasjon på en B2B nettjeneste kan gi stort utslag på kundeopplevelsen i tjenesten. Han konkluderer også at kundeservice er spesielt viktig i tilfeller hvor informasjonen ikke er god, og kunder sliter med å finne fram. (McLean 2017, 665). Dette går også igjen i en studie gjort ved Ulster University in 2003, også her konkluderer de med at kundeservice kan være enda viktigere enn selve teknologien (Yang, Humphreys og McIvor 2003)

2.3 B2B E-commerce

For å stille bedre forberedt til å utvikle løsningen måtte vi også se nærmere på hva som gjør en B2B e-commerceplatform en suksess. Ifølge en studie gjort av Pierre Berthon et al. (2008) har markedet du opererer i mye å si for resultatet av en B2B e-commercesatsing. De har funnet at det er tre viktige punkter man må forholde seg til:

- Landets verdier
- Korrupsjon (tillit)
- E-readiness (infrastruktur som legger til rette for eller står i veien for å utføre en digital handel)

Vår oppdragsgiver jobber primært i det norske markedet hvor det er lav korrupsjon, høy tillit og god infrastruktur for denne typen handel (Berthon et al. 2008, 87). Vi kan derfor forvente at innføring av en e-commerceplatform vil åpne nye muligheter. Noen av disse mulighetene har allerede blitt kartlagt i en studie av Polina Fauska, Natalia Kryvinska og Christine Strauss (2013):

- Man kan spare tid og penger ved bruk av en slik plattform under innkjøpsprosesser
- Bedre kommunikasjonsflyt
- Både kjøper og selger vil kunne bli mer produktive

I en artikkel skrevet av Yadong Huang, Yueting Chai, Yi Liu og Jianping Shen (2019) blir mangelen på personlig tilpasning for hver enkelt bruker dratt frem som et problem med dagens e-commerceløsninger. Dette er relevant i lys av større plattformer rettet mot ulike kundegrupper, hvor det kan være relevant å segmentere hvilket produkter som er tilgjengelig for kunder basert på deres behov.

3 Prosess og metode

I dette kapittelet går vi gjennom hvilke valg vi har tatt når det kommer til prosjektmetodikk og hvilke verktøy vi har valgt å bruke. Vi diskuterer flere forskjellige alternativer og begrunner våre valg.

3.1 Utviklingsmetode

Vi har valgt å bruke Scrum. I dette avsnittet forklarer vi hva Scrum er, hvorfor vi har brukt det og hvordan vi har brukt det. Vi begrunner valget og diskuterer hvordan vi valgte prosjektstyringsprogram og hvorfor.

3.1.1 Scrum

Scrum ble lansert som prosjektmetodikk av Jeff Sutherland og Ken Schwaber (Rubin 2013, xxxv). Det er en prosjektmetodikk som bygger på kontinuerlige leveranser av brukbar kode. Scrums mål er å adressere noen av de største problemene man møter på i løpet av en utviklingsprosess, nemlig at usikkerhet er umulig å unngå og at det er umulig å vite alle kravene til et system på forhånd (Sutherland 2007, 1). Scrum prøver å løse disse problemene ved å gjøre utviklingsprosessen mer dynamisk og åpen for endringer underveis.

Scrum metodikken lener seg på noen grunnprinsipper:

- **Backlog** er listen over krav til prosjektet. I starten av et prosjekt kan dette typisk være krav fra oppdragsgiver. Etterhvert som utviklingen kommer i gang kan disse kravene endre seg og backloggen kan blir større eller mindre. (Rubin 2013, 18)
- **Sprint** er et ordtak som beskriver en periode i utviklingen. En Sprint skal ha en tidsbegrensning og den skal ha et sett med mål. En Sprint kan vare alt fra en uke til en måned. (Rubin 2013, 20)
    - For hver Sprint skal det opprettes en egen Sprint backlog. I denne Sprint backloggen kan man plukke ting fra prosjekt backloggen som skal gjøres i løpet av Sprinten.

- **Daily Scrum** er et 15 minutters møte man skal ha hver dag. Her skal hvert gruppemedlem svare på hva hen har gjort, hva hen skal gjøre idag og hvilke problemer som kan hindre hen fra å få utført dagens oppgaver. (Rubin 2013, 23)
- **Sprint review** er et møte man har på slutten av en Sprint der gruppen diskuterer hvordan Sprinten gikk. Her skal man vurdere hva som ble gjort. Man skal i løpet av dette møtet også gjennom en aktivitet som heter **Sprint Retrospective.** Her skal man vurdere prosessen istedenfor produktet (Rubin 2013, 27)

3.1.2 Hvorfor valgte vi Scrum?

Før vi begynte prosjektet diskuterte vi hvilken metodikk vi ønsket å gå for. Det var viktig for oss å velge en metodikk som gir oppdragsgiver en innsikt i utviklingsprosessen. Vår oppdragsgiver hadde ikke oversikt over all funksjonalitet de trengte i den endelige løsningen. En del av vår jobb ville derfor være å jobbe sammen med oppdragsgiver underveis for å avdekke hvilke behov de hadde. Vi vurderte flere metodikker men etter noen runder med diskusjon sto vi igjen med to kandidater, Kanban og Scrum. Vi så noen fordeler ved å bruke Kanban, siden vi kunne bruke enklere prosjektstyringsverktøy og brukt mer tid på bare arbeid. Det ville også gitt oppdragsgiver enkel mulighet til å følge med på hva vi holdt på med, men det ville ikke hatt den samme ‘røde tråden’ gjennom arbeidet slik man får gjennom Scrum. Vi endte til slutt opp med å velge Scrum. Vi så for oss at det kunne ta litt mer tid å komme seg inn og at det krevde litt mer tid gjennom møtene i Scrum, men at det ville være verdt det siden prosjektet var så stort og det var viktig å holde seg organisert. Scrum lar oss nemlig dokumentere det meste av arbeid i et strukturert og ryddig format. Sprintene ville også gi oss konkrete mål å jobbe mot fra uke til uke. Som vi nevnte tidligere ville kravene til prosjektet vårt fort kunne endre seg underveis. Siden Scrum legger til rette for og forventer dette var det en ekstra fordel. Samtlige gruppemedlemmer har også gjennom studiet gjennomført et prosjekt over to semestre der vi lærte å ta i bruk Scrum. Dette bidro til at læringskurven ble kortere.

3.1.3 Vår implementasjon av Scrum

If various combinations of extra or no pickles, tomatoes, lettuce, cheese, and so on can lead to over a million ways to make a hamburger, there must be billions of possible ways to implement Scrum. (Cohen 2013, xxxi)

Scrum er laget for å kunne tilpasses brukerne. For noen vil sikkert månedslange Sprinter være perfekt, mens for andre er det mer optimalt med kortere Sprinter på en arbeidsuke. Vi måtte derfor ta en del valg på starten av prosjektet der vi definerte hvordan vi skal bruke scrum:

- Vi bestemte oss for å legge Sprintene til en arbeidsuke. Dette gjorde vi fordi vi ville ha hyppige leveranser slik at vi kontinuerlig kunne vurdere prosessen og produktet.
- Siden det ikke ble praktisk mulig for hele gruppen å møte bedriften etter hver uke, lagde vi en ordning der gruppen sammen reflekterte og vurderte Sprintresultatene for å så la våre produkteier ta med seg dette videre til oppdragsgiver/stakeholders.
    - Hver andre eller tredje Sprint møttes hele gruppen, inkludert produkteier og stakeholders, til et felles møte.
- Vi valgte å ta i bruk Azure DevOps som prosjektstyringsverktøy. Dette skal vi diskutere nærmere i neste avsnitt.
- I Daily Scrum valgte vi heller å ha en åpen dialog mellom gruppemedlemmene i stedet for de tre standardspørsmålene definert i Scrum.

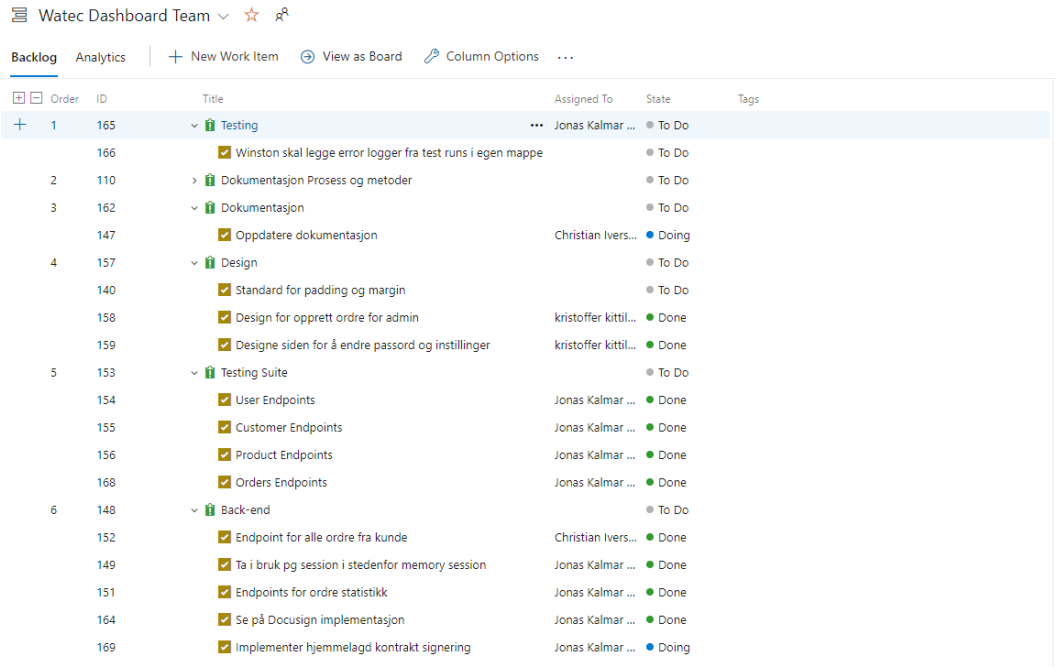
3.1.4 Prosjektstyringsverktøy

Skal man bruke Scrum er man avhengig av å ha et system som kan holde styr på f.eks produkt backlog, Sprint backlog og hvem som gjør hva. Vi måtte også belage oss på å gjennomføre prosjektet på hjemmekontor, da koronapandemien gjorde det umulig å møtes. Siden vi ikke hadde mulighet til å møtes ble det enda viktigere å ha en god struktur og bra planlegging. Vi begynte derfor å se etter programmer som kunne hjelpe oss med å oppnå dette, og vi fant noen alternativer:

- Scrumwise
- Azure DevOps

- Jira

Scrumwise hadde vi erfaring med fra tidligere prosjekter, men vi følte at det var litt primitivt. Jira og Azure var gratis for små grupper mens Scrumwise koster penger. Vi valgte derfor å droppe Scrumwise da vi ikke mente det var verdt prisen. Både Jira og Azure hadde mange nyttige egenskaper som for eksempel Git og pipelines. Vi endte opp med å velge Azure DevOps fordi vi kunne teste flere av funksjonene gratis. Det var også enkelt å sette opp pipelines og Git-integrasjon slik at vi kunne samle både kode, dokumentasjon og Sprint planlegging på samme plass.



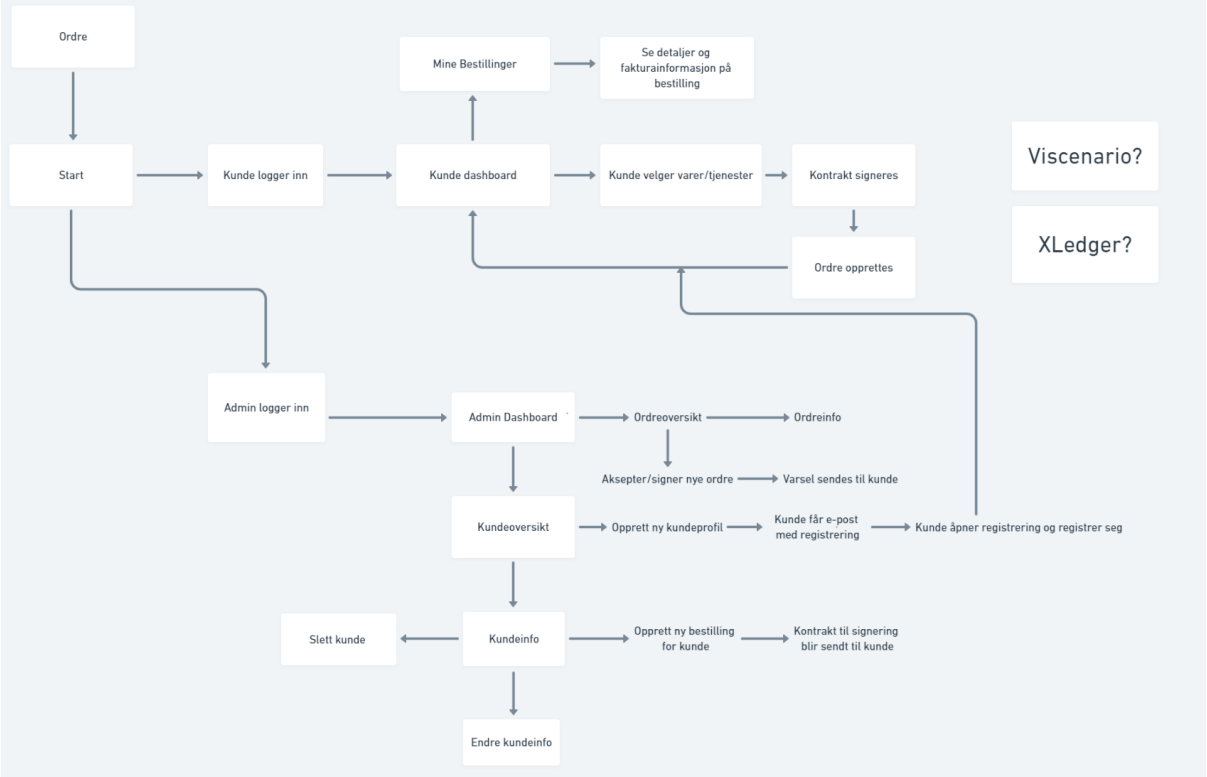
Figur 1 - Skjermdump fra product backlog i Azure DevOps.

Azure DevOps ga oss mange funksjoner som gjorde hjemmekontor mye enklere.

- **Pipelines** - Vi kunne sette opp automatiske handlinger hvis ny kode ble publisert, f.eks: Når vi er klare til å laste opp ny kode til produksjonsserver kan DevOps automatisk kjøre de automatiske testene for å sjekke om koden er klar til å legges på server.
- **Kanban** - Board til hver Sprint hvor vi kunne se hva alle på gruppen gjorde til enhver tid.
- **Git** - Integrasjon som lot oss laste opp koden vår til DevOps og dermed samle alt på en plass.

3.2 Prosjektplan

Etter at vi hadde valgt en prosjektmetodikk gikk vi i gang med planleggingen av prosjektet. Vi startet med å utarbeide en prosjektbeskrivelse. I prosjektbeskrivelsen definerte vi kravene fra oppdragsgiver og hvilke funksjoner vi trengte å implementere. For komplett prosjektbeskrivelse, se vedlegg A.



Figur 2 - Bilde av flowchart for løsningen.

Da prosjektbeskrivelsen var ferdig lagde vi et flowchart hvor vi oversatte kravspesifikasjonen til et “kart” med funksjoner. Her fikk vi et overblikk over hva vi trengte å gjøre framover, og det hjalp oss med å senere lage produktbackloggen.

3.2.1 Risikoanalyse

På starten av prosjektet så utarbeidet vi en risikoanalyse for å være bedre forberedt dersom vi skulle møte på hindringer underveis. Denne analysen skulle hjelpe oss vurdere hvor sannsynlig ulike scenarioer som kunne ha et negativt utslag på prosjektet var Denne analysen ligger under vedlegg B

3.2.2 Dokumentasjon

I prosjektet vårt har vi brukt flere forskjellige former for dokumentasjon. Vi har hatt Sprint Reviews hvor vi skriver ned hva som har blitt gjort hver uke av programmering og dokumentasjon, i tillegg til Retrospectives for å vurdere vår egen bruk av Scrum. Vi har også referater av møter med bedrift og veileder. Dette har vi lagret i Google Drive og Azure DevOps. I Azure DevOps har vi laget vår egen wiki hvor vi har skrevet ned informasjon om løsningen. Dette er informasjon som komponenter, sider, database modeller og hvordan det skal brukes.

3.3 Datainnsamling

3.3.1 Brukertester

Vi har tatt i bruk brukertesting som et verktøy for å innhente tilbakemeldinger på applikasjonen. Denne tilbakemeldingen kan så brukes for å prosjektere behov for videreutvikling eller endringer i systemet. Brukertestene ble gjort som en del av Undersøkelsesmetoderfaget vi hadde tidligere i våren, med intensjon om å utføre forskning relevant til vår egen bacheloroppgave. For å utføre brukertester har vi tatt utgangspunkt i “Moderert Fjerntesting” fra boken ‘Praktisk Brukertesting’ av Eli Toftøy-Andersen og Jon Gunnar Wold. Dette går ut på at brukeren får tilgang til en løsning som skal testes. Brukeren får også et sett med oppgaver som vedkommende løser mens testleder ser på via skjermdeling. Vi valgte denne måten fordi det var den eneste metoden som var praktisk mulig på grunn av koronapandemien. Fjerntesting har også en potensiell fordel over tradisjonell brukertesting siden brukeren kan sitte i sitt eget hjem og gjennomføre testene, som kan gjøre at testen føles mindre “kunstig” enn hvis den blir gjort på et eget testlokale vi har satt opp for dem. (Toftøy-Andersen og Wold 2012, 260).

Testrunden ble gjort med fem personer, som ifølge den anerkjente brukervennelighetsforskeren Jakob Nielsen er et ideelt antall personer for å utføre en test. Dette er ikke fordi fem brukere er nok til å avdekke alle feil; i følge hans forskning vil du kun avdekke 80% av feil med fem deltakere, men du vil derimot ikke avdekke 100% av feil før du har testet med hele 15 personer. Det vil si at det er sterk

avtagende avkastning på å teste flere deltakere enn fem, og du vil få mye repetisjon av svar fra deltakere. Nielsen mener derfor det er bedre å gjøre mindre tester med fem deltakere så mange ganger som mulig som en del av en iterativ designprosess, enn å gjøre færre store tester med 15 deltakere for å avdekke alle feil på en gang. (Nielsen 2000)

Vi tok også i bruk System Usability Scale (SUS) for å innhente kvantitativ data fra brukertestene og få evaluert hvordan vår løsning kan måles opp mot andre IT-systemer. SUS fungerer ved å gi brukeren et spørreskjema med ti punkter etter avsluttet test. På skjemaet får brukeren oppgave å rangere de forskjellige påstandene fra én til fem. Deretter regner man ut svaret basert på om svaret er et partall eller oddetall. Oddetallsspørsmål regnes ut ved at man trekker fra en på svaret til brukeren, mens partallspørsmål regnes ut ved å starte med fem og trekke fra svaret til brukeren. Til slutt kan man gange summen av alle resultatene med 2.5 og få en rangering fra 1 - 100. (Sauro 2011)

Brukertestene ble gjort i henhold til de etiske retningslinjene vi lærte i PJ6100, og all data hentet ut ble anonymisert. Samtykkeerklæringen vi sendte til testdeltakere i forveien av testen ligger under vedlegg C

3.4 Verktøy

I dette kapittelet presenterer vi hvilke verktøy, programmeringsspråk og rammeverk vi har tatt i bruk.

3.4.1 Programmeringsspråk

Da vi skulle velge programmeringsspråk var det flere faktorer å ta hensyn til. Vi måtte først og fremst gjøre en vurdering på gruppens kompetanse og hvilke språk, teknikker eller rammeverk vi kunne lære i løpet av prosjektet. Vi sto mellom C# og JavaScript. Argumentene for C# var å bruke det sammen med Azure Cloud. Vi hadde lite erfaring med både Azure og C# og vi syntes det var vanskelig å finne gode informasjonskilder for nybegynnere. Derfor falt valget på JavaScript. Alle på

gruppa har god kontroll på språket og vi kjenner til flere rammeverk og biblioteker til dette språket, som f.eks. Express og React.

JavaScript er innebygd i alle moderne nettlesere og kan dermed være et selvinnlysende alternativ når man bygger applikasjoner som kjører på web. I tillegg kjører de fleste nettlesere også JavaScript just-in-time (JIT). Dette innebærer at koden ikke må kompileres før den blir kjørt. (Clark 2017)

Vi valgte å ta i bruk Node.js for vår backend. Node.js er miljø som lar deg kjøre JavaScript kode utenfor nettleseren. Vi valgte Node.js fordi det er godt dokumentert, ledende i industrien og flere store selskaper som f.eks Twitter og PayPal har tatt det i bruk. (SimilarTech 2021)

3.4.2 Rammeverk

JavaScript har et stort utvalg rammeverk og biblioteker som kan brukes for å spare tid og få et høyere abstraksjonsnivå på noen av de tyngre oppgavene, som f.eks. å ta i mot og lese forespørsler til server, lagring av data og oppretting av gjenbrukbare komponenter. I dette delkapittelet skal vi gå gjennom hvilke rammeverk vi har valgt og hvorfor vi har valgt dem.

3.4.2.1 Frontend

React er et JavaScript-bibliotek for å bygge brukergrensesnitt og UI-komponenter. Hovedhensikten til React er å forenkle tilstandsstyring og å rendre ut tilstanden på nettsiden automatisk når den endrer seg. Det er hovedsakelig utviklet og vedlikeholdt av Facebook. React er i dag det mest brukte frontend-rammeverket i Javascript og tas i bruk av mange av de største teknologiselskapene i verden, for eksempel Uber, Netflix og Amazon. (Węglarz, 2020). I lys av dette, og at React er det rammeverket vi selv har mest erfaring med, var det naturlig å ta det i bruk i prosjektet.

React byr på mange funksjoner for å lage et dynamisk grensesnitt der informasjonen kan endres uten å laste inn siden på nytt. React kan derimot ikke hjelpe med oss

med å lage et responsivt grensesnitt. Derfor har vi valgt å ta i bruk CSS-biblioteket Bootstrap. Bootstrap er et bibliotek som inneholder en del ferdiglagde grensesnitt-komponenter man kan ta i bruk. Det kommer også med posisjoneringsverktøy som gjør det enkelt å lage et grensesnitt som fungerer på alle plattformer. Ulempen med Bootstrap er at designet fort kan bli nesten identisk med andre Bootstrap-sider. Det krever derfor at vi må bruke mye tid på å "personalisere" designet slik at det blir unikt.

3.4.2.2 Backend

Da vi skulle planlegge backenden vår måtte vi finne ut av to ting: Hvordan skal vi lagre dataen, og hvordan skal vi kommunisere med databasen? Vi vurderte å ta i bruk en skytjeneste som Firebase til å håndtere databasen og autentisering, men vi valgte det bort siden Firebase bruker en NoSQL database fremfor en SQL database. Vi skal diskutere forskjellene i neste avsnitt.

Da vi skulle velge database måtte vi vurdere om vi skulle ta i bruk en NoSQL eller en SQL database. De to typene skiller seg ved at en SQL database bygger på relasjoner; en rad i databasen kan inneholde en nøkkel som peker til en annen rad i en annen tabell i databasen. NoSQL derimot bygger ikke på relasjoner og bruker ikke tabeller for å lagre data. Dette lar deg sette inn data som JSON-objekter.

Når man skal velge mellom de to kommer det an på hva du skal bruke det til og hvilken type data du skal lagre. Vi trengte primært å lagre data om produkter, ordre og kunder. Denne dataen inneholder en del relasjoner, for eksempel:

- En kunde kan være tilknyttet ingen eller mange ordre
- En ordre kan være tilknyttet ett eller flere produkter og den er tilknyttet én kunde
- En kontrakt skal være tilknyttet én ordre, og en ordre kan være tilknyttet ingen eller én kontrakt.

Siden det meste av vår data bygger på relasjoner valgte vi å gå for en relasjonsdatabase. Vi valgte PostgreSQL fordi kildekoden er åpen, den er gratis å bruke og har et stort antall følgere.

Etter valget av database måtte vi finne ut hvordan vi skulle lage bindeleddet mellom databasen og webklienten, også kalt webserveren. Node.js har et innebygd HTTP-bibliotek vi kunne tatt i bruk. Vi valgte heller å ta i bruk et bibliotek som heter Express. Express bygger på Node.js sitt HTTP bibliotek og tilbyr tilnærmet lik funksjonalitet uten at man må “lese” alle forespørsler manuelt. Det sparer oss for veldig mye tid; vi kan med Express opprette et nytt endepunkt på under fem minutter.

3.4.3 Server

Vi trengte en server som koden vår kunne kjøre på da den var ferdig. Denne serveren trengte ikke nødvendigvis å være den oppdragsgiver ender opp med å bruke, men vi ønsket å kunne ha ferdig kode kjørende på nettet slik at oppdragsgiver kunne gå inn å følge med på utviklingen underveis.

Før vi satt i gang for å lete etter server hadde vi noen krav:

- Den må kunne kjøre Node.js
- Den må ha støtte for å hoste en PostgreSQL-database
- Den bør helst ha Git-integrasjon

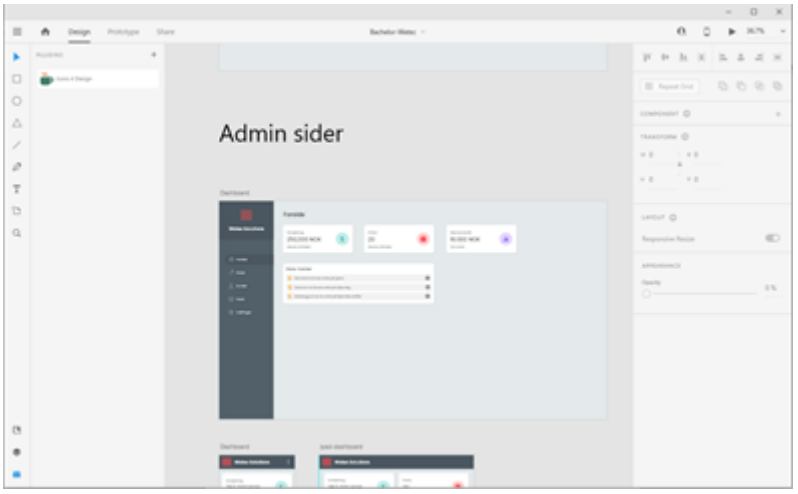
Etter å ha undersøkt litt fant vi frem til Heroku. Heroku er en skyplattform som tilbyr hosting av applikasjoner skrevet i en rekke språk, inkludert Node.js. Koden kjører i en “dyno” som er en slags egen boks, og Heroku håndterer oppdatering av programvare slik at vi kan fokusere på utvikling av koden. Den støtter også PostgreSQL og det er enkelt å skalere opp serveren om oppdragsgiver har behov for mer kapasitet i fremtiden. Den har også full støtte for både Git, GitHub og Pipelines som gjorde at den huket av alle boksene på kravlisten vår.

3.4.4 Designverktøy

For å utarbeide grensesnittet til løsningen var vi avhengige av å ta i bruk et designverktøy. Vi trengte et verktøy som lot oss lage wireframes og prototyper før vi

satt i gang med kodingen. Vi kikket på flere alternativer som Framer, Figma og Adobe XD.

En av de høyest prioriterte funksjonene vi trengte var muligheten til å kunne redigere samme dokument. På grunn av den pågående koronapandemien må vi benytte hjemmekontor og det er derfor viktig for oss å kunne samarbeide effektivt når vi skal designe over nett. Alle de tre alternativene tilbyr dette.



Figur 3 - Skjermdump av Adobe XD.

Etter å ha sett nærmere på de tre ser vi at de har mange av de samme funksjonene. Vi likte ikke grensesnittet i Invision og sto derfor igjen med Figma og Adobe XD. Da vi undersøkte forskjellene på de to så fant vi en artikkel skrevet av Thomas Urlikas hvor han går igjennom hvordan de gikk frem da de skulle velge et nytt verktøy. Her har de sammenlignet Figma, Sketch, Adobe XD og Framer X. Adobe XD vinner en knapp seier med 0.5 poeng over Figma (Urlikas, 2020). Dette forteller oss at Adobe XD og Figma er nesten helt identiske når det kommer til funksjonalitet. Vi endte med å velge Adobe XD fordi vi foretrakk grensesnittet og fordi vi har litt erfaring med å bruke det fra før. Adobe XD har også et stort utvalg med nyttige plug-ins vi kan ta i bruk.

4 Analyse og utforming

4.1 Diskusjon med bedriften

For å utvikle kravspesifikasjon brukte vi mye tid på å intervjue bedriften om dagens salgsprosess, hvilket steg i prosessen som var mest tidkrevende for dem og hvordan de så for seg at en digital salgsplattform kunne vært utformet for dem. Vårt overordnet mål var å digitalisere og forenkle bestillingsprosessen for Watec og deres kunder uten at det skulle føre til store endringer i flyten eller kreve mye opplæring. For å best finne ut av hvordan alt fungerte i dag brukte vi også et par dager på å se på ansatte registrere og behandle kundebestillinger og intervjue de samme ansatte i etterkant om hvordan de gjør det og hva de syntes var vanskelig. En gjenganger var at det ofte var mye frem og tilbake før en bestilling kunne føres inn. Ofte sendte en kunde en mail med litt oversikt over hva de så etter, så måtte en ansatt svare med hvilket tjenester/produkter de kunne tilby og spørsmål om hva kunde ønsket. Kunden ga deretter uttrykk for hva og hvor mye de ønsket å bestille, som bedriften måtte svare med et formelt tilbud og vente på bekreftelse av kunden. Ifølge flere ansatte vi diskuterte dette med var dette ofte tilfellet også med stamkunder som hadde bestilt fra dem før. Selv om de visste noenlunde hvilke tjenester de ønsket, siden de hadde fått det utført av Watec tidligere, var de ofte ikke komfortable med å sende en bestilling på de tjenestene rett ut før de hadde hentet inn bekreftelse på at Watec kunne levere det og prisen de ville betale. Oppdragsgiver ga uttrykk for at den fram-og-tilbake metoden med bestillinger ikke alltid var nødvendig, siden tjenestene deres allerede utføres etter priskalkyler.

4.2 Krav til løsning

Basert på diskusjon vi hadde med ansatte i bedriften og gjennom dialog med produkteier kom vi opp med en kravspesifikasjon som begge parter var enige ville ha nytteverdi for bedriften, og som samtidig var overkommelig for vår gruppe med de tekniske ferdighetene vi satt med og tidsrammen til prosjektet.

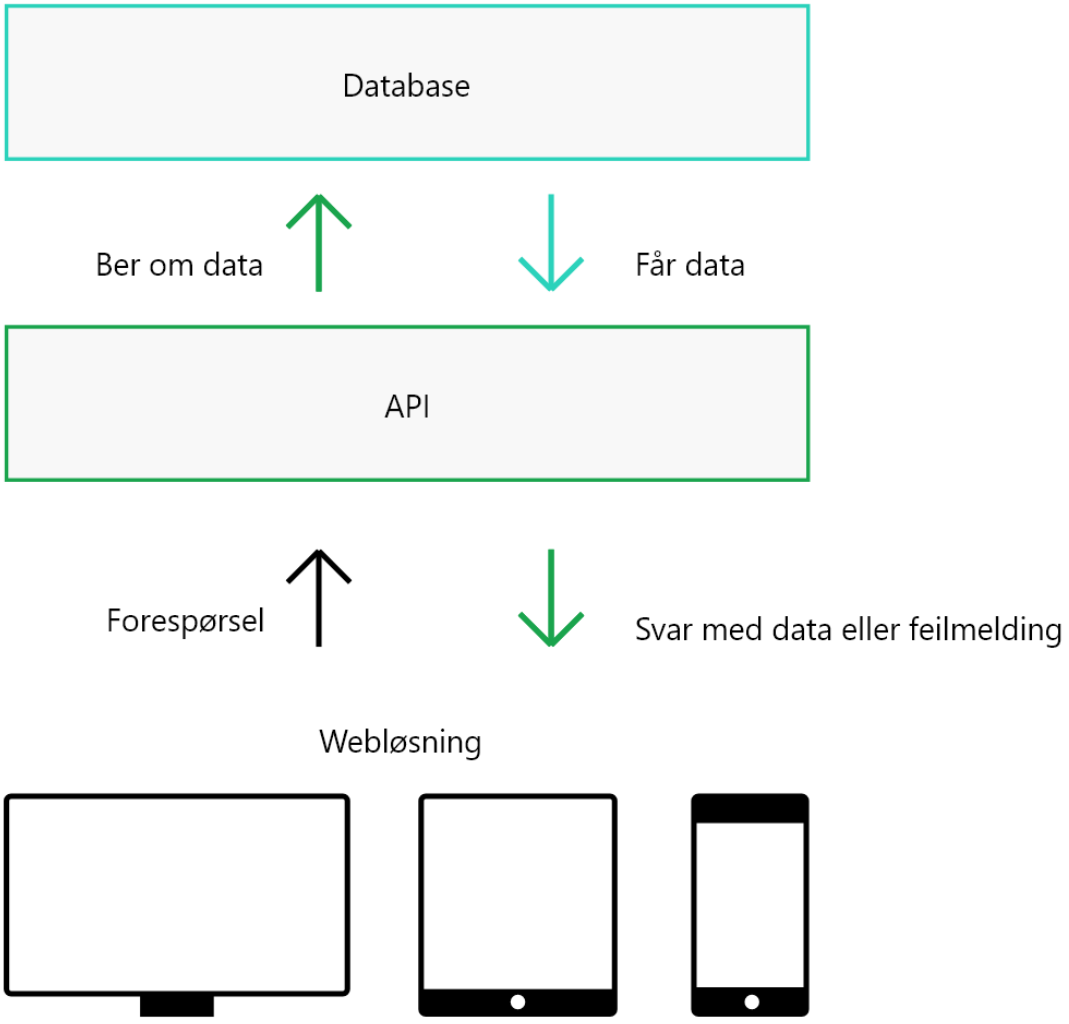
Funksjonalitet	Må ha	Burde ha	Fint å ha
Skal utvikles på webplattform og støttes av nettlesere	X		
Admin må kunne invitere kunder til siden for dem	X		
Statistikk som gir verdifull innsikt om kunder på plattformen		X	
Kunder skal kunne registrere ordre av tilgjengelige varer	X		
Nettsiden må ha et funksjonelt og brukervennlig design	X		
Løsningen kan enkelt videreutvikles for nye bruksområder			X
Admin skal kunne opprette bestillinger på vegne av kunder		X	
Kontraktsignering for ordre med BankID eller Docusign		X	
Integrere tjenester sånn som Visma og Viscenario via API			X
Admin og kunder kan søke gjennom tidligere ordre	X		
Løsning må fungere på PC, tablet og mobil		X	
Lagerbeholdning kan registreres og endres av admin			X
Servicehistorikk på maskiner i maskinpark			X

Tabell 2 - Prioritert Kravsliste

I tillegg til funksjonaliteten på kravslisten var det viktig for Watec at selskapet kunne implementere løsningen uten at det kom på bekostning av de viktige kundeforholdene de hadde i dag. Det var derfor ekstra viktig at brukervennligheten i systemet var god, i hvert fall på sidene kundene ville interagere med. Hvis systemet oppfylte alle funksjonskravene men ikke var brukervennlig eller tidsbesparende for kunder ville ikke Watec kunne ta i bruk systemet, da gjentagende kunder er ekstremt viktig for selskapet, slik nevnt i innledningen.

5 Teknisk løsning

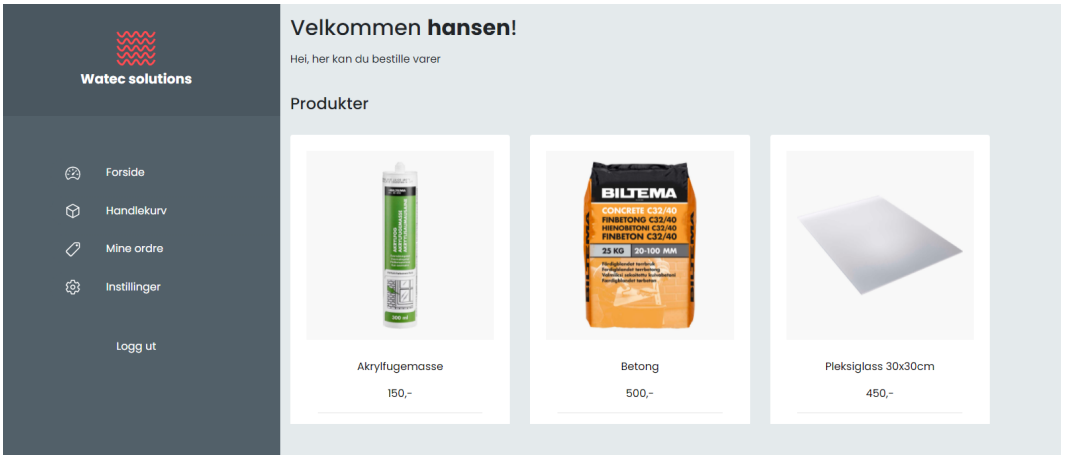
5.1 Teknisk Presentasjon



Figur 4 - Figur som viser hvordan løsningen henger sammen

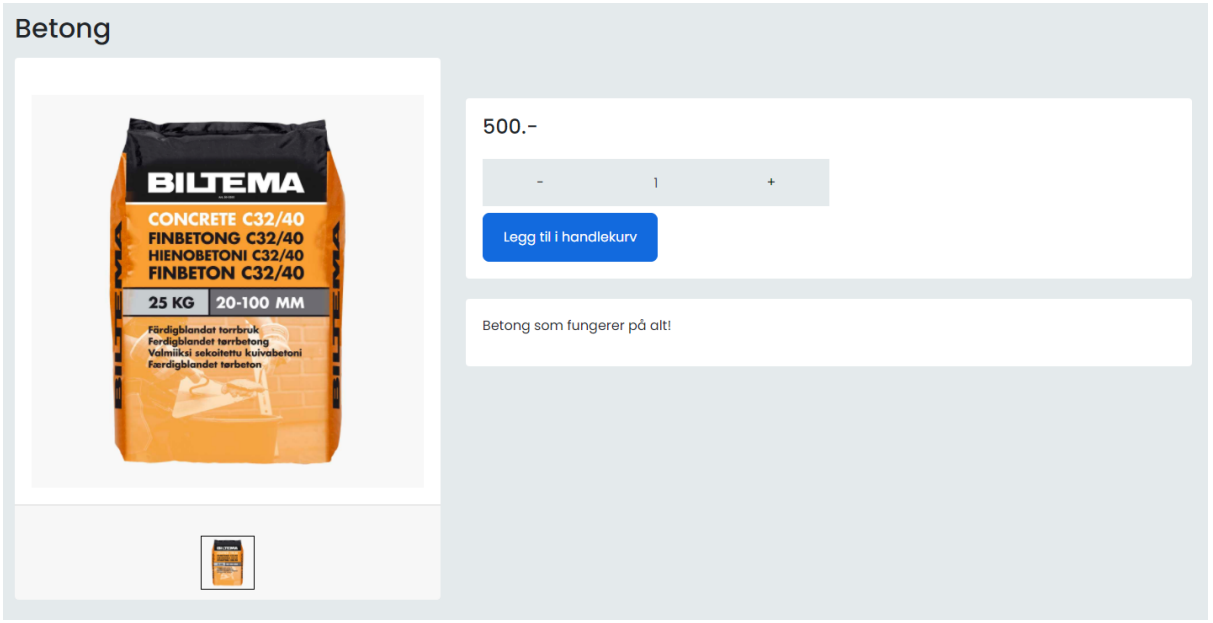
Vi har laget en e-commerceplatform for vår oppdragsgiver som har som mål å flytte ordreprosessen fra e-post til et strukturert system. Plattformen vår lar oppdragsgiver invitere kunder og velge hvilke produkter de skal ha tilgang til. Når kunden lager en bruker kan de bestille disse produktene og se status på dem. Vi kan dele opp løsningen i tre deler: Webklienten som kunder og ansatte bruker, databasen som lagrer dataene og APIet som fungerer som en bro mellom databasen og klienten.

5.1.2 Kundeportal



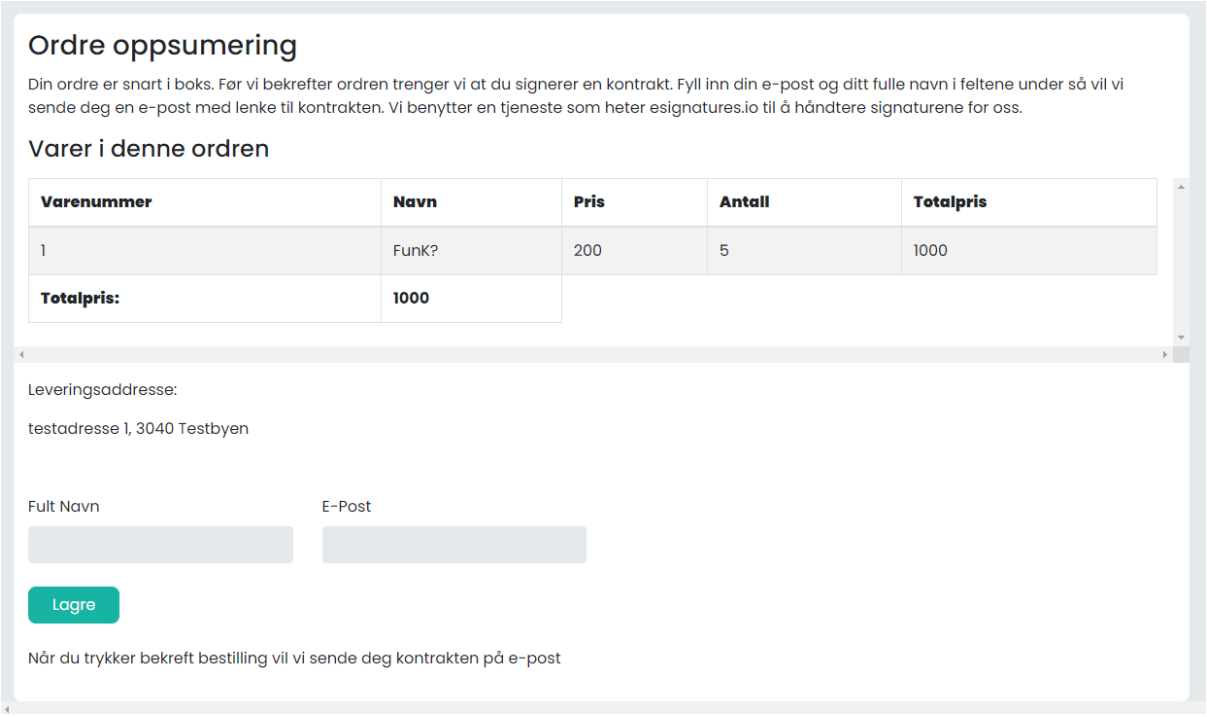
Figur 5 - Skjermdump av kundeportal.

Når kunden logger seg inn i systemet blir den møtt med kundeportalen. Herfra kan kunden finne produkter, tjenester og tidligere ordre. Kunden kan trykke seg inn på hvert enkelt produkt for å finne mer informasjon om produktet og kan også bestille dette produktet. Etter kunden har bestilt vil ordren vises under “mine ordre”.



Figur 6 - Skjermdump av produktsiden

På produktsiden kan kunden legge produktet og ønsket antall i handlekurven. Her kommer det også mer informasjon om varen, og én eller flere bilder.



Figur 7 - Skjermdump av ordreoppsummeringen.

Når kunden er ferdig med å legge til varer vil kunden bli sendt til oppsummeringen hvor man må fylle inn navn og e-post til personen som skal signere kontrakten. Etter kunden har gjort dette og trykket lagre vil de bli sendt videre til oversikten for den ordren, og kontrakten sendes via e-post for signering.

Figur 8 - Skjermdump av "mine ordre"-siden.

Når kunden går til 'mine ordre'-siden vil de få en oversikt over alle bestillinger i en tabell. Her kan kunden se gjennom sine ordre og trykke seg inn på hver enkelt ordre for å få mer informasjon.

Figur 9 - Skjermdump av ordredetaljer-siden.

Dette er oversikten for ordre 1. Her finner kunden all relevant informasjon rundt denne ordren, slik som:

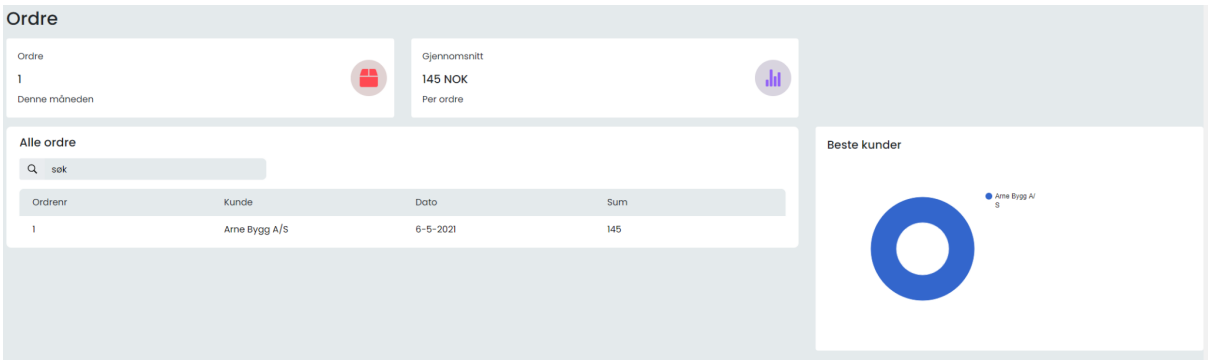
- Lenke til signert kontrakt i PDF-format, dersom kunden har allerede signert
- Antall produkter i ordren og prisen på disse
- Hvilken leveringsadresse som er registrert på ordren
- Eventuelle kommentarer på ordren
- Sendingsstatus

5.1.3 Administratorportal

Figur 10 - Skjermdump av administratorportal.

Fra administratorportalen kan administratorer se statistikk fra den siste måneden og en oversikt over siste hendelser i systemet. Man har også tilgang på en rekke sider der man kan se:

- Oversikt over alle ordre, varer og kunder
- Innstillingerside der man kan endre passord på kundebrukere og opprette nye administratorbrukere
- Invitasjons side der man kan se, opprette og trekke tilbake invitasjoner til nye kunder

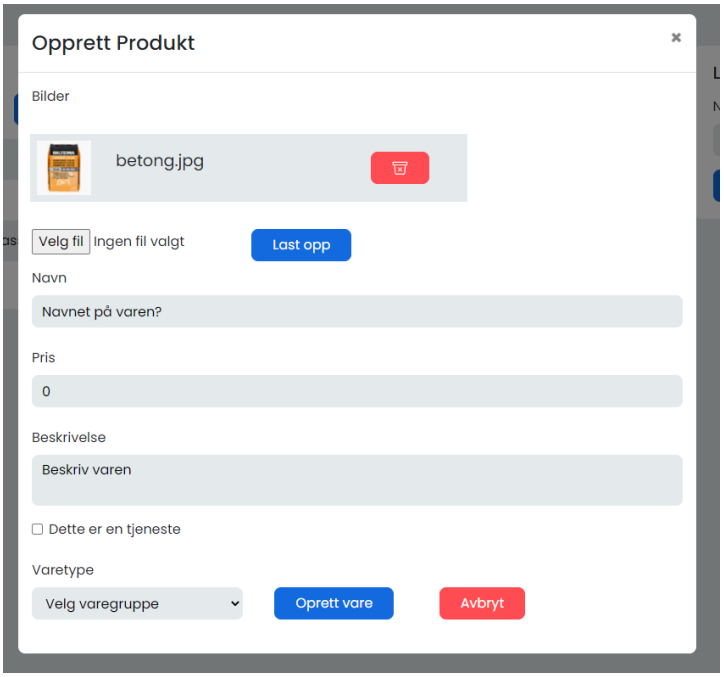


Figur 11 - Skjermdump fra “alle ordre”-siden.

Fra ‘alle ordre’-siden kan man se en oversikt over alle ordre og statistikk over hvilke kunder som har bestilt og betalt mest. Man kan også se gjennomsnitt og antall ordre denne måneden. I ettertid så vi at vi kunne lagt til en funksjon for å se statistikk fra tidligere måneder.

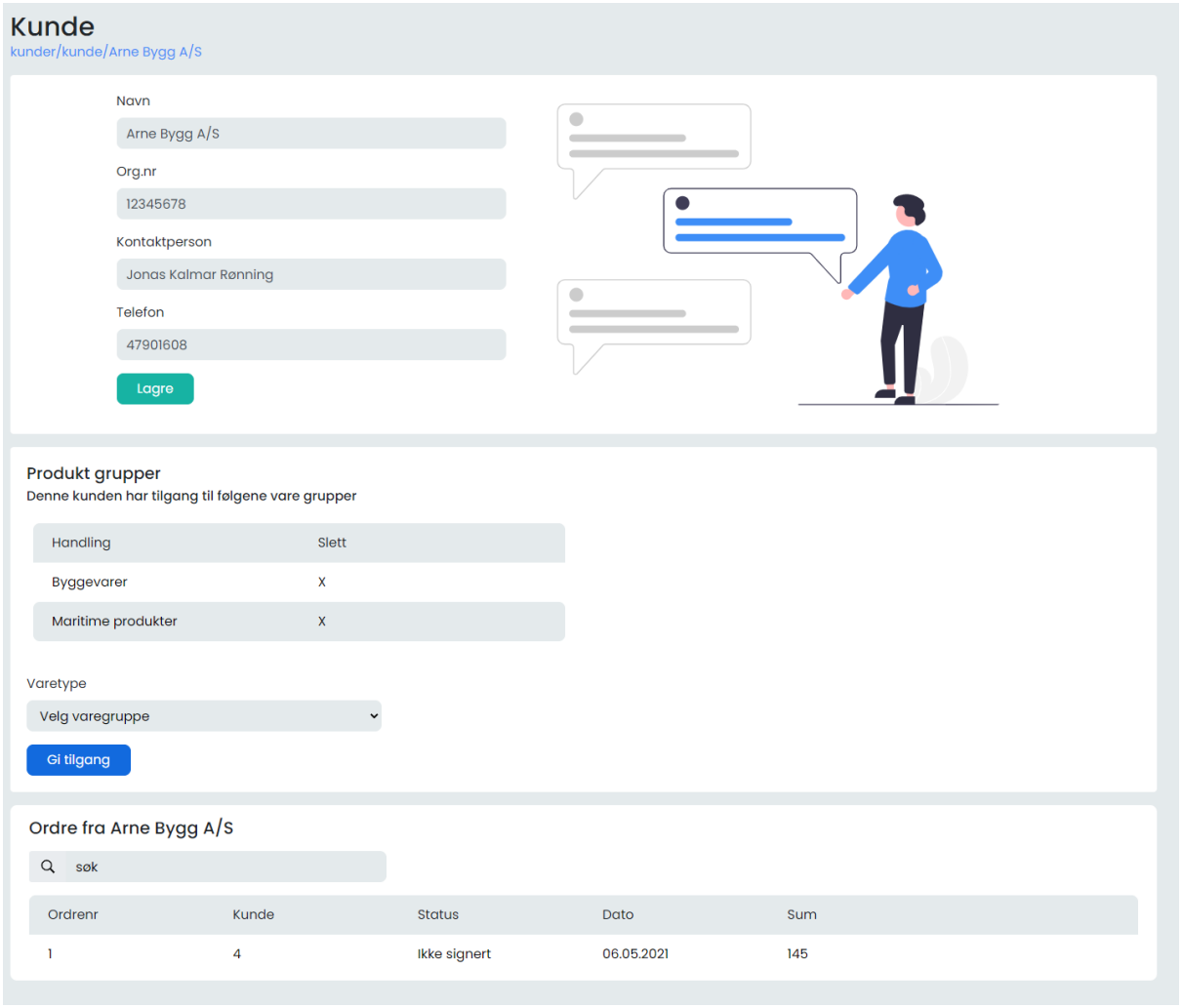
Figur 12 - Skjermdump fra "varer"-siden.

Fra 'varer'-siden kan man opprette nye varer og varegrupper. Man kan også trykke seg inn på en vare for å se all informasjon om varen og redigere den. En svakhet på denne siden er mangelen på en funksjon for å slette varegrupper. Dette er fordi hvert produkt krever å være tilknyttet en varegruppe for å kunne eksistere. Det ville derfor bydd på mye arbeid å fjerne en varegruppe helt. Man kan derimot arkivere produkter slik at de ikke vises for kunder. Denne løsningen åpner for at man i fremtiden kan skjule varegrupper som kun er tilknyttet arkiverte produkter.



Når man skal opprette en vare må man fylle ut relevant informasjon og tildele varen en varegruppe. Herfra kan man også laste opp bilder av varen.

Figur 13 - Skjermdump fra "ny vare"-siden.



Figur 14 - Skjermdump av kundeinformasjonsiden.

Fra kundesiden kan man se all relevant informasjon om kunden:

- Kontaktinformasjon
- Hvilke produktgrupper denne kunden har tilgang til
- Alle ordre tilknyttet denne kunden

Her kan man også endre kontaktinformasjon og velge hvilke varegrupper denne kunden skal ha tilgang til. En funksjon som ikke er vist på dette bildet er også muligheten for å opprette en ny ordre på kunden. Denne funksjonen eksisterer, men knappen for å gå til den mangler på dette bildet.

5.1.4 Infrastruktur

Serveren vår tar i bruk flere forskjellige teknologier for å kommunisere med webløsningen. Vi skal gå gjennom de to viktigste teknologiene.

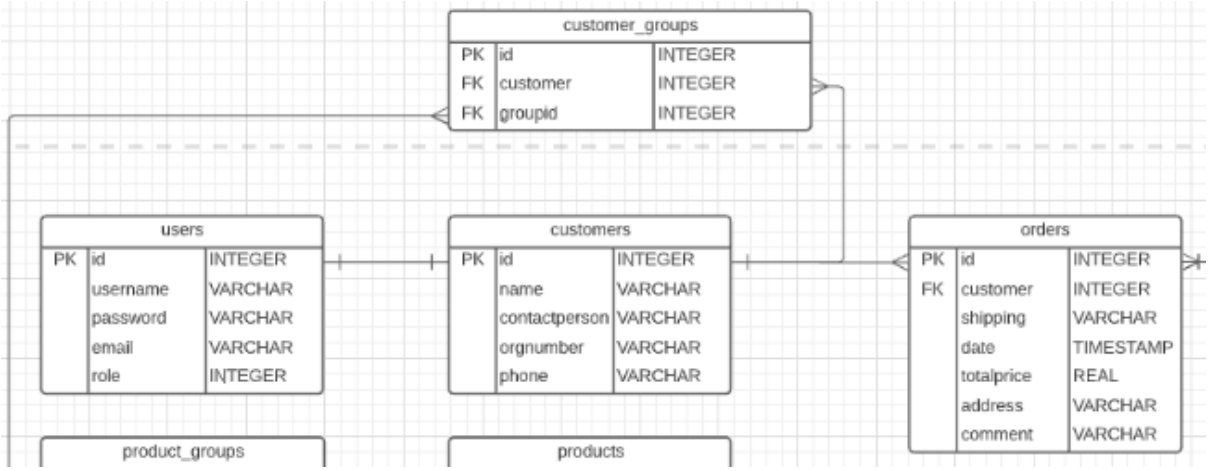
5.1.4.1 API

For vår backend har vi valgt å ta i bruk et RESTful API for å håndtere kommunikasjon mellom server og klient. Et RESTful API fungerer ved å tilby et sett med ressurser. En ressurs kan for eksempel være et bilde, et dokument eller en tjeneste. Et annet karakteristisk trekk ved RESTful API er at APIet ikke husker hvem som har forespurt hvilken ressurs (Fielding 2000, 76).

Figur 15 - Strukturen til en request mot APIet.

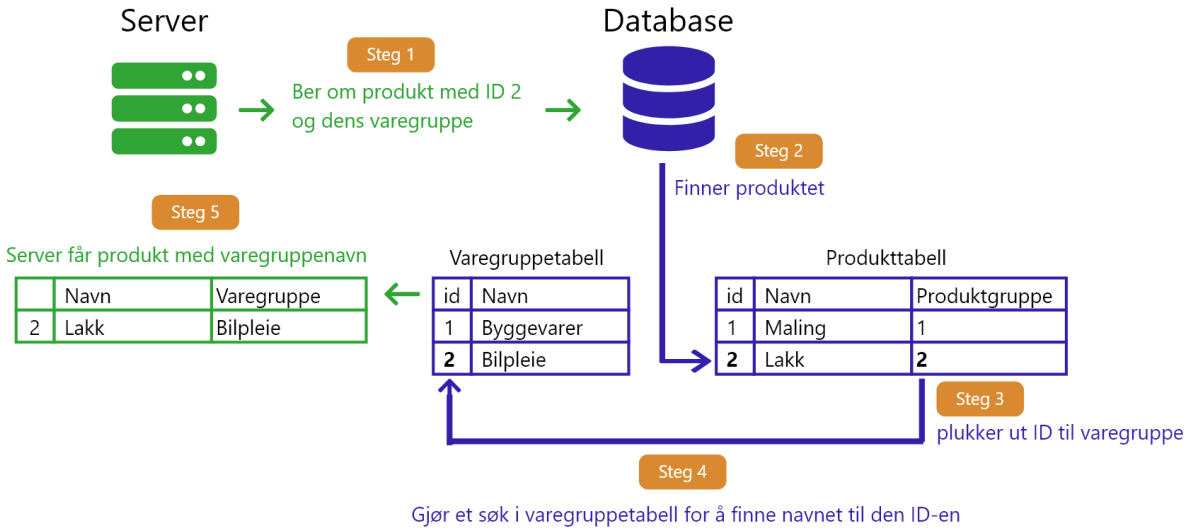
På bildet over ser man et eksempel på en forespørsel til vårt API. Den starter med å spesifisere forespørseltypen som er GET (hente ut informasjon) etterfulgt av adressen til ressursen. Denne består av lokasjonen til ressursen som i dette tilfellet er “customers”, deretter identifikasjonen på den spesifikke ressursen. Med andre ord ber vi om kunde nummer 24.

5.1.4.2 Database



Figur 16 - Skjermdump av UML-skjemaet for databasen.

Vi har som nevnt tidligere tatt i bruk en relasjonsdatabase. Denne fungerer ved at radene i databasen kan ha relasjoner til hverandre. For eksempel vil alle produkter ha en varegruppe. En varegruppe kan tilhøre flere produkter. Ved å strukturere databasen på denne måten slipper vi å lagre samme type data to ganger.

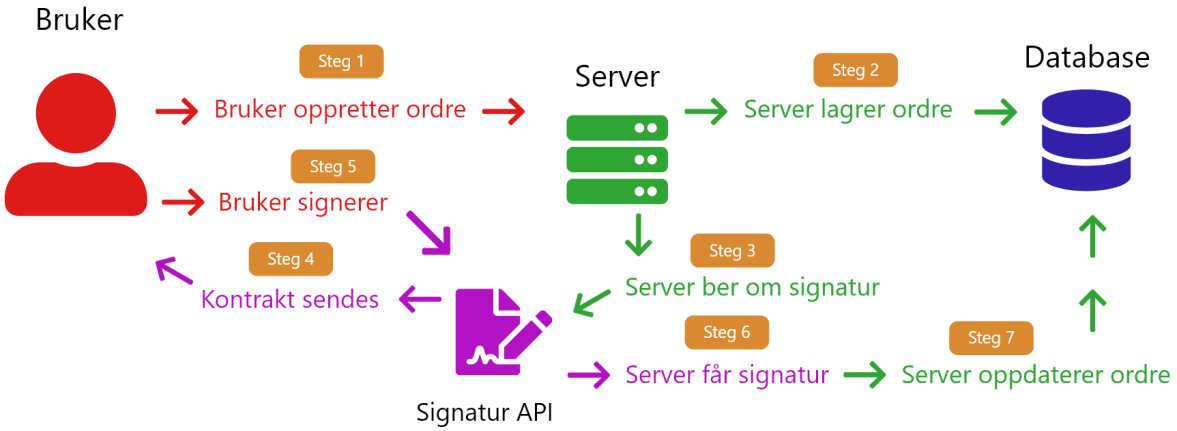


Figur 17 - Illustrasjon av en databasespørring.

Bildet over viser hvordan en spørring i en relasjonsdatabase kan se ut. Serveren ønsker å få et produkt og navnet på varegruppen. Siden navnet ligger i en annen tabell må de to tabellene slås sammen. Bildet over viser at “Lakk” er tilknyttet produktgruppe “2”. Dette betyr at databasen kan finne navnet på produktgruppen ved å slå opp raden med ID 2 i produktgruppetabellen. Om produktgruppenavnet

skulle bli endret vil databasen fortsatt slå opp på raden med ID 2 og finne det nye navnet.

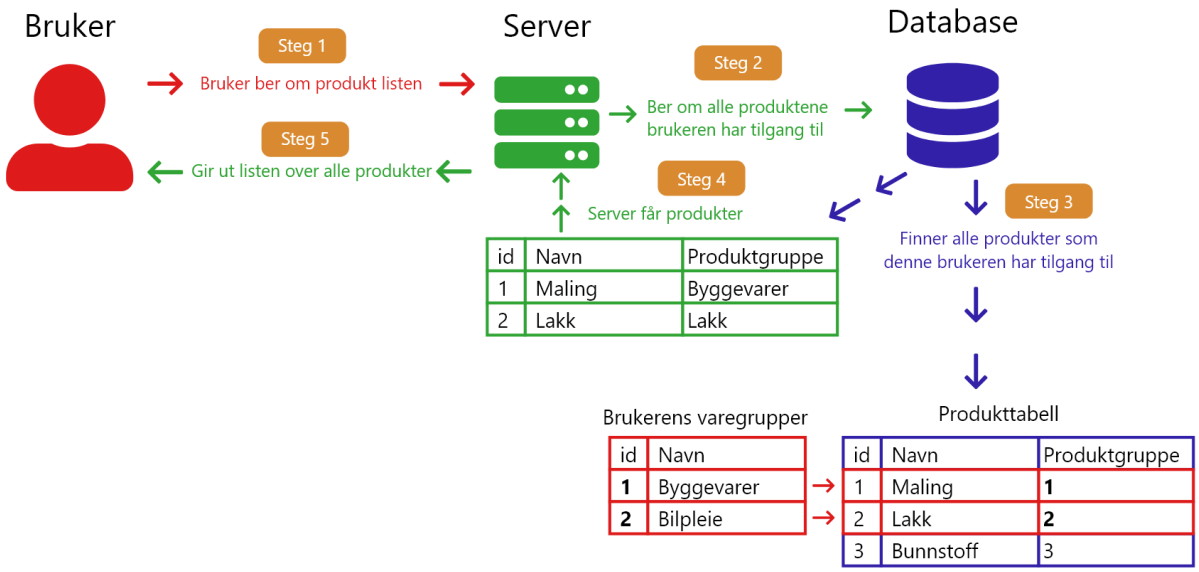
5.1.5 Ordreprosess



Figur 18 - Visualisering av ordreprosessen.

På bildet over ser du hvordan ordreprosessen fungerer etter at en bruker har lagt inn en ordre. Den starter med at serveren oppretter ordren og markerer den som usignert. Deretter vil serveren sende en beskjed til signatur APIet om at en kontrakt må sendes. Signatur APIet vil deretter sende kontrakten til kunden på e-post og gi serveren beskjed når kunden har signert. Når kunden har signert kontrakten vil serveren endre ordren fra usignert til signert og lagre en lenke til signert kontrakt slik at kunde og admin kan se kontrakten på ordresiden. Vi valgte å bruke et tredjeparts API for signeringen da vi ikke var helt sikre på hvordan man skulle implementere en slik løsning juridisk riktig. Vi tok i bruk eSignatures.io sitt API for å løse dette, det var fordi de hadde en ryddig dokumentasjon og var billigere enn mange andre på markedet som bruker subscription-modeller. Med eSignatures betaler man kun $0.4 per signerte dokument, dersom man ikke skal signerer mange hundre dokumenter i måneden og antall signeringer som blir gjort vil variere fra måned til måned kan dette være rimeligere enn subscription-modellene.

5.1.6 Kundegrupper



Figur 19 - Illustrasjon som viser hvordan server og database kommuniserer for å hente produkter.

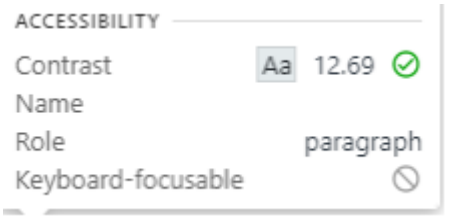
Et av kravene som dukket opp underveis var å kunne begrense hvilke produkter hver enkelt kunde hadde tilgang til. Vi løste dette med et system som bygger på at hver bruker har tilgang til et sett varegrupper. En varegruppe kan være tilknyttet ingen eller mange produkter.

Når en bruker ber om å få listen over produktene vil serveren spørre databasen om å få alle produktene som er tilgjengelig for denne kunden. Databasen vil automatisk sjekke hvilke varegrupper denne brukeren har tilgang til. Deretter vil den svare med produktene som er tilknyttet disse varegruppene. Dette er en av de store fordelene med å velge en SQL-database siden den tillater slike avanserte spørringer.

5.2 Design og Utforming

Figur 20 - Fargepalett.

Før vi begynte designprosessen måtte vi søke inspirasjon fra andre løsninger for å se hvordan vi best kunne utforme et design for en e-commerce webløsning. Vi landet til slutt på seks farger med god kontrast til hverandre som skulle være bærebjelken i designet. Flere av fargene tar utgangspunkt i Bootstrap sitt fargeutvalg. Bootstrap er et bibliotek som gir tilgang på mange ferdiglagde grensesnittkomponenter. Problemet med Bootstrap er at nettsider laget med Bootstrap ofte kan bli veldig like. Vi valgte derfor å lage våre egne implementasjoner ved å blande og mikse forskjellige Bootstrap-komponenter slik at vi fikk et unikt design.



Figur 21 - Contrast-rating som vist i Google Chrome

Gjennom prosjektet har vi brukt Google Chrome sitt kontrastsjekker verktøy. Dette verktøyet gir oss en rangering på kontrast. Rangeringen baserer seg på kontrastnivået. En vanlig tekst bør være minimum 4.5:1. (w3.org,2018)

Verktøyet gir oss en god rangering mellom 12 og 17 poeng på de viktigste komponentene som tabeller, input-felter og navigasjonsmeny. Dessverre så vi noen svakheter i designet; noen av knappene våre fikk en dårligere rangering ned mot seks og syv poeng. Til tross for dette ligger løsningen vår mellom WCAG nivå AA og AAA. WCAG er et et sett med retningslinjer laget for å gjøre webløsninger mer tilgjengelige og definerer noen suksesskriterier som gir deg et nivå på tilgjengelighet fra A til AAA. (w3.org,2021)

5.4 Dokumentasjon

`GET /api/products`

Returns a list of all products stored in the database.
Alternatively, if withGroup is sent with the body, returns the list alongside the names of the productgroups they are assigned to.

Body

`withGroup: boolean`

Returns

`401 Unauthorized: User is not logged in`
`204 No Content: No products are stored in the records`
`200 OK: Returns a list of products, alongside their groupname depending on the withGroup body parameter`

Figur 22 - Utdrag fra dokumentasjonen, definerer endpoint som henter ut alle produkter.

Dokumentasjon til kodebasen er skrevet i wikien til Azure DevOps, hvor Git repositoryet også oppbevares. Vi har vurdert dokumentasjon som en viktig del av den tekniske løsningen da den skal være en del av det vi gir bedriften når arbeidet er over, og vil være vesentlig informasjon dersom bedriften skal videreutvikle plattformen.

```
/** Edit a order with a spesified id
 *
 * @param products the products in this order
 * @param customer the customer this order belongs to
 * @param signed is this order signed?
 * @param id the id of the order
 */
export const editOrder = async (products : IProductOrder[], cus
```

Figur 23 - Skjermdump fra JSDoc-kommentarer i koden.

Vi har vært spesielt nøye med å dokumentere APIet og databasen slik at andre utviklere enkelt kan bygge nye løsninger på plattformen. I tillegg til å dokumentere APIet har vi også dokumentert de gjenbrukbare komponentene som vi har laget for brukergrensesnittet. Vi har også inkludert gode kommentarer i koden til backenden

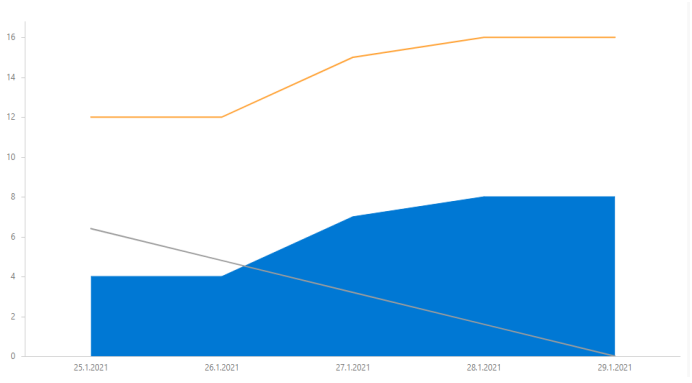
for å beskrive hva de ulike funksjonene gjør. Vi har tatt i bruk JSDoc-syntaksen på disse kommentarene.

6 Diskusjon

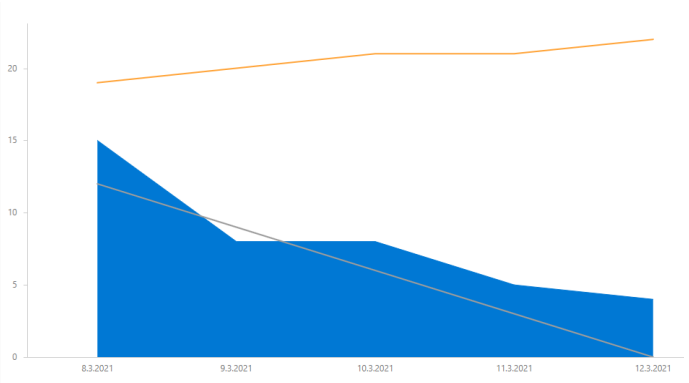
I dette kapittelet skal vi diskutere resultatene fra vårt bachelorprosjekt. Vi skal gå gjennom og drøfte våre metoder og teknisk resultat. I tillegg skal vi se på nytteverdien av vår løsning for vår oppdragsgiver og diskutere hvordan løsningen kan videreutvikles i fremtiden.

6.1 Vurdering av metoder, prosess og gjennomføring

6.1.1 Vurdering av Scrum



Figur 24 - Burndowngraf Sprint 1

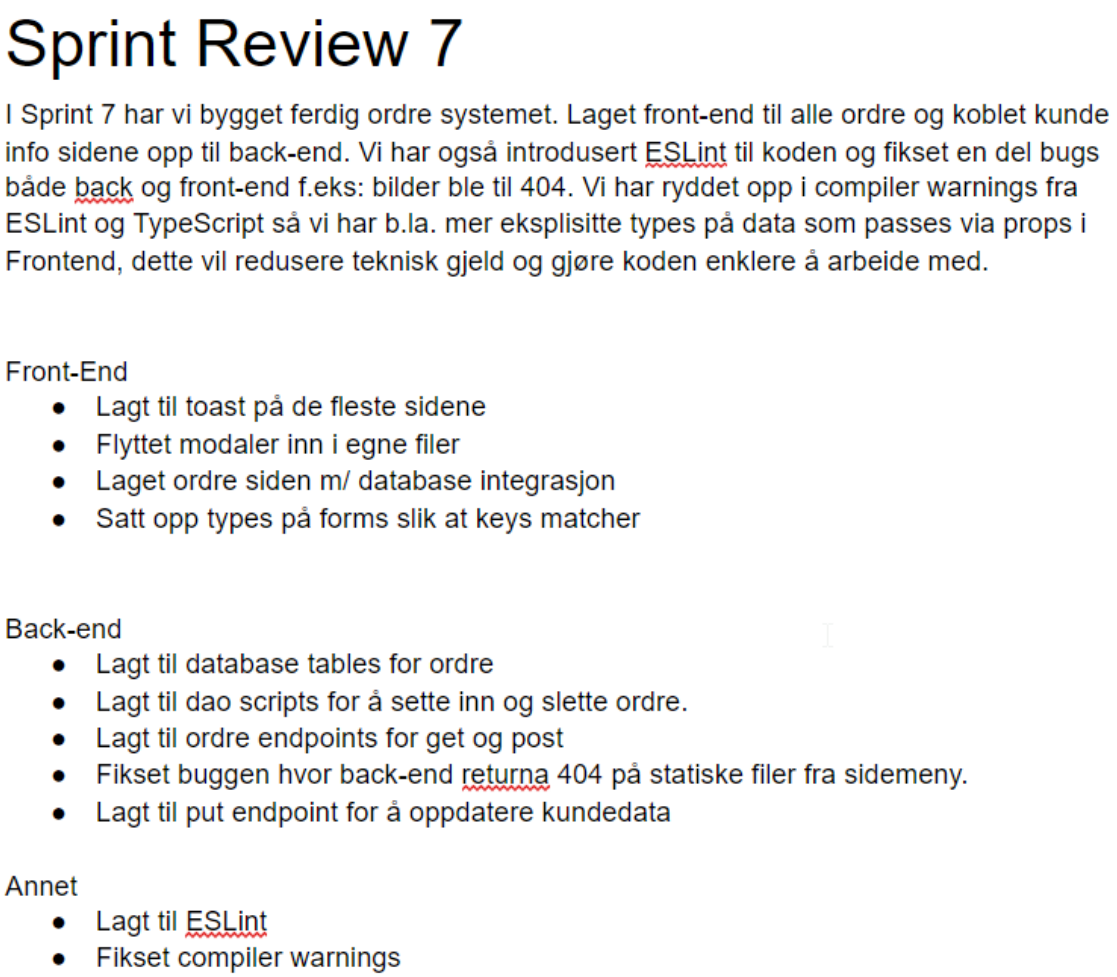


Figur 25 - Burndowngraf Sprint 8

Vi valgte som sagt å ta i bruk Scrum som prosjektmetodikk. Et av de viktigste punktene i Scrum er estimering; man estimerer hvor mye arbeid en oppgave krever for å fullføre. Etterhvert som man fullfører Sprinten får man en “burndowngraf” som representerer hvor mye arbeid man har fullført. I starten av Sprinten var vi ganske dårlige på estimering, og dette gjenspeiles i grafen til venstre. Etter en del Sprinter ble vi gradvis bedre på dette og mot Sprint åtte og utover begynte vi å få en jevnere graf.

Vi har også tatt i bruk møtene som anbefales av rammeverket til Scrum. Vi startet hver dag med en kopp kaffe hvor vi pratet litt sosialt om ting utenfor prosjektet i noen minutter før vi gikk i gang med en Daily Scrum. I Daily Scrum diskuterte vi hva hver på gruppen hadde planlagt å jobbe med den dagen, og om noen ville trenge hjelp

med noen arbeidsoppgaver. Dette tok oss 3-15 minutter, varierende av hvor avanserte arbeidsoppgaver vi hadde den dagen. Deretter satte vi i gang med arbeid. Gjennom arbeidsdagene satt vi oftest sammen på Discord slik at vi kunne enkelt diskutere om det dukket opp problemer underveis, men et par dager ble vi sittende offline på grunn av Internettproblemer eller forstyrrelser i omværelsen.



Sprint Review 7

I Sprint 7 har vi bygget ferdig ordre systemet. Laget front-end til alle ordre og koblet kunde info sidene opp til back-end. Vi har også introdusert ESLint til koden og fikset en del bugs både back og front-end f.eks: bilder ble til 404. Vi har ryddet opp i compiler warnings fra ESLint og TypeScript så vi har b.la. mer eksplisitte types på data som passes via props i Frontend, dette vil redusere teknisk gjeld og gjøre koden enklere å arbeide med.

Front-End

- Lagt til toast på de fleste sidene
- Flyttet modaler inn i egne filer
- Laget ordre siden m/ database integrasjon
- Satt opp types på forms slik at keys matcher

Back-end

- Lagt til database tables for ordre
- Lagt til dao scripts for å sette inn og slette ordre.
- Lagt til ordre endpoints for get og post
- Fikset buggen hvor back-end returna 404 på statiske filer fra sidemeny.
- Lagt til put endpoint for å oppdatere kundedata

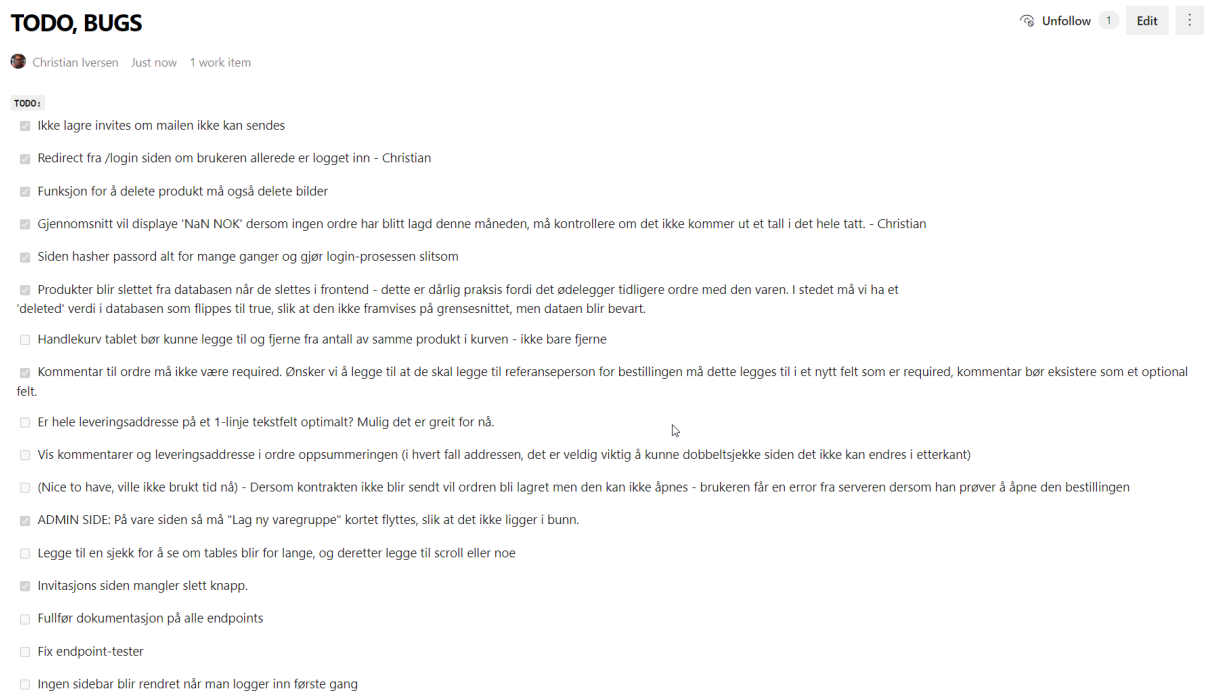
Annet

- Lagt til ESLint
- Fikset compiler warnings

Figur 26 - Utkast fra Sprint Review.

På slutten av hver Sprint passet vi på å ta Sprint Reviews og Retrospectives. Reviewene hjalp alle holde seg oppdatert på hvilke endringer som hadde skjedd på løsningen. Dette var spesielt nyttig de gangene gruppemedlemmer jobbet på helt andre deler av appen og kunne få en oppsummering av hva som skjedde andre steder. Vi fikk eksempelvis brukt reviews til å gå gjennom hvilke endpoints som hadde blitt bygget ferdig. Dette var spesielt hjelpsomt når et medlem som jobbet med frontenden skulle ta i bruk ulike endpoints uten å selv ha skrevet noe kode i APIet. Retrospektivene ble brukt til å vurdere vår egen bruk av Scrum og hvilket

forbedringer som var mulig. Gjennom disse Retrospektivene fikk vi sett mer på hvordan vi estimerte arbeidsoppgavene, og fant ut at vi slet mye med ‘scope creep’, samt å avgrense vår ‘definition of done’. ‘Scope creep’ førte til at vi endte opp med å legge til nye arbeidsoppgaver utenfor det vi egentlig arbeidet med, og siden vi ikke fikk avgrenset vår ‘definition of done’ var det oppgaver som ble markert som ferdig et par Sprinter på rad selv om det var mer arbeid å gjøre. Etter disse Retrospektivene ble vi flinkere til å sette faste arbeidsoppgaver og finne enighet om hva vår ‘definition of done’ skulle være, som også kommer fram i burndowngrafene vi viste over.



Figur 27 - Skjermdump av to-do liste.

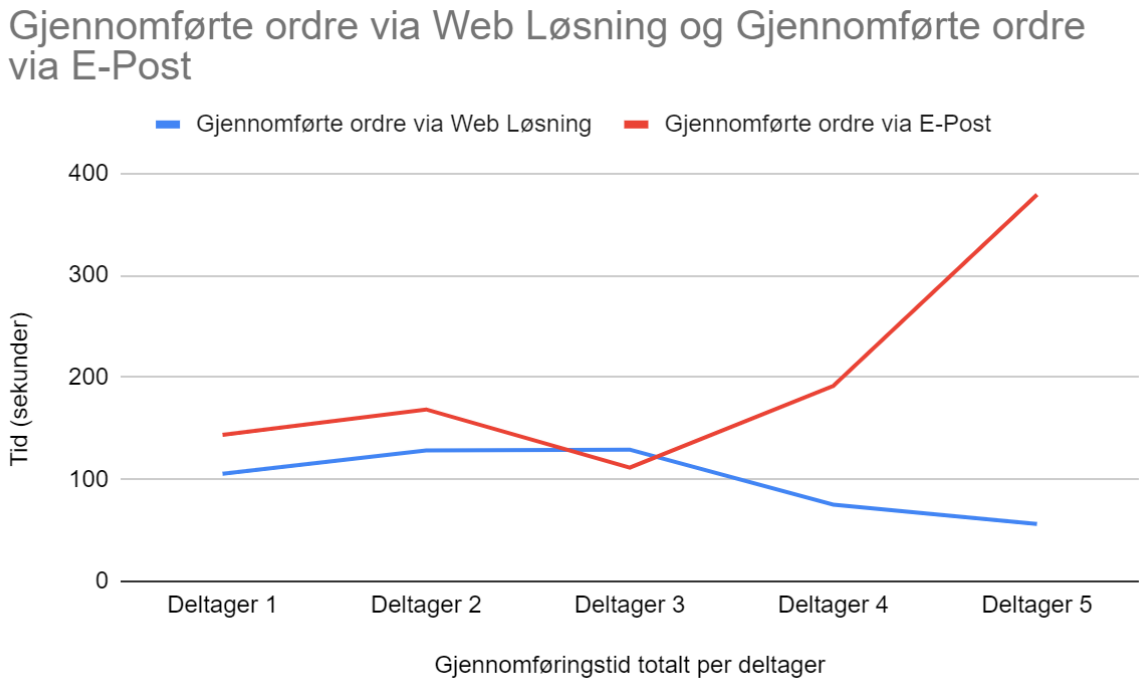
Mot slutten av prosjektet måtte vi derimot vike litt fra Scrum. I de siste tre dagene vi jobbet med produktet hadde vi samlet opp en del bugs og små endringer som måtte korrigeres før vi var klar til å levere produktet til kunden. Vi bestemte oss derfor for å ikke bruke en Sprint på denne delen, men heller jobbet ut ifra en to-do liste som vi hadde kommentert på siden starten av prosjektet. Vi tok dette valget fordi det var såpass mange små ting som måtte gjøres som vi alle skjønte hvordan skulle gjøres, men som vi ikke helt rakk å formulere inn i en ordentlig Sprint. I denne siste innspurten hadde vi også møter med bedrift og veileder, samtidig som vi skrev mye i dokumentasjonen. Derfor ble det mye fram og tilbake på arbeidsoppgaver og vi fikk ikke samme arbeidsflyt som vi var vant til i de tidligere Sprintene. I etterkant tror vi at

dette var et greit kompromiss for å få fullført det som måtte på plass, og vi mener det kan vise at Scrum-rammeverket ikke alltid må følges slavisk for at utviklingen skal være produktiv, men at fleksibilitet kan være nødvendig i enkelte situasjoner.

6.1.2 Brukertesting

6.1.2.1 Sammenligning av løsninger

Brukertestene vi gjorde som en del av Undersøkelsesmetoderfaget ga oss nyttig tilbakemelding om applikasjonen og hva som kunne forbedres. I tillegg testet vi applikasjonen vår opp mot en lignende bestillingsprosess gjort med e-post, tilnærmet lik slik Watec i dag tar i mot bestillinger. Hensikten var å se hvilken effekt løsningen kunne utgjøre på kundeopplevelsen. Watec hadde selv ikke noe erfaring med kunde- og ordrebehandling på nettbaserte plattformer, så de var interesserte i at vi skulle undersøke om løsningen kunne gi fordeler for deres kunder. Basert på resultatene fra brukertestene kunne vi dra noen interessante konklusjoner om løsningen. Først så vi at completion rate, som måler hvor mange av oppgavene brukerne fikk utført, lå på 100% på både løsningen vår og den originale e-post metoden. Basert på dette antar vi at selve bestillingsprosessen er såpass enkel at brukere vil kunne løse det uansett, sett at de får et forståelig grensesnitt. Derimot så vi et større sprik på hvor lang tid det tok å gjennomføre oppgavene. På nettløsningen greide brukerne å finne et produkt, finne en ordre og gjennomføre en hel bestilling på 26.95 sekunder i gjennomsnitt, mens de samme oppgavene tok brukerne på e-post 47.91 sekunder, altså 20.96 sekunder lengre. Vi ser på dette som en indikasjon at brukere syntes det var enklere og mer oversiktlig å bruke nettløsningen, noe som også går igjen på resultatene fra SUS-skjemaene. Her ga e-postgruppen dårligere score på kompleksiteten av løsningen, og svarte at systemet var mer ulogisk.



Figur 28 - Gjennomføringstid på webplattformen og e-post.

Etter brukertestene ble fullført gjorde vi også intervjuer med testdeltakerne, også her fikk webløsningen vår mye bedre tilbakemelding. De fleste deltakerne på webløsningen hevdet at de ville vært komfortable med å bruke den løsningen siden de hadde brukt mange lignende e-commerce plattformer tidligere, og oppsettet mente de var veldig likt. Deltakerne på e-post følte at det å gjøre selve bestillingen over e-post ville vært OK sett at de pratet med et annet menneske, men de syntes det var unødvendig vanskelig å finne fram til produktliste som var sendt til dem på tidligere mail. Selv når de fant produktlisten sa noen at de ville sendt mail til bedriften for å bekrefte produktutvalg og pris før de hadde bestilt noe, en observasjon vi selv så bedriften gjøre med mange av sine kunder i starten av prosjektet, selv om de mente det var tungvint å måtte sende flere mail bare for å gjøre én bestilling. Vi mener dette kan ha med måten informasjonen tolkes på, siden akkurat samme pris og produktinfo ble vist på e-post som på web. En teori vi har på hvorfor dette skjer er at kunder vet at nettsider enkelt kan oppdateres, de stoler derfor på at det som står på en nettside alltid er oppdatert og relevant. En e-post som ble sendt til dem tidligere derimot er uforanderlig, det er ingen måte for selskapet å endre informasjonen her uten å sende en helt ny e-post. Denne observasjonen er relevant i lys av forskningen til McLean vi refererte til i kapittel 2, som sa at kvaliteten på

informasjon gir stort utslag på kundeopplevelsen, her ser vi kanskje at måten informasjon presenteres på også gir utslag på hvordan kvaliteten tolkes av brukere. Vi mener i alle fall at dette er enda en indikasjon på at kundeopplevelsen kan bli forbedret ved å ta i bruk webplattformen vi har utviklet.

Som nevnt ble disse dataene generert fra brukertester utført som en del av vår eksamen i PJ6100 Undersøkselsesmetoder.

6.1.2.2 Tilbakemeldinger på applikasjon

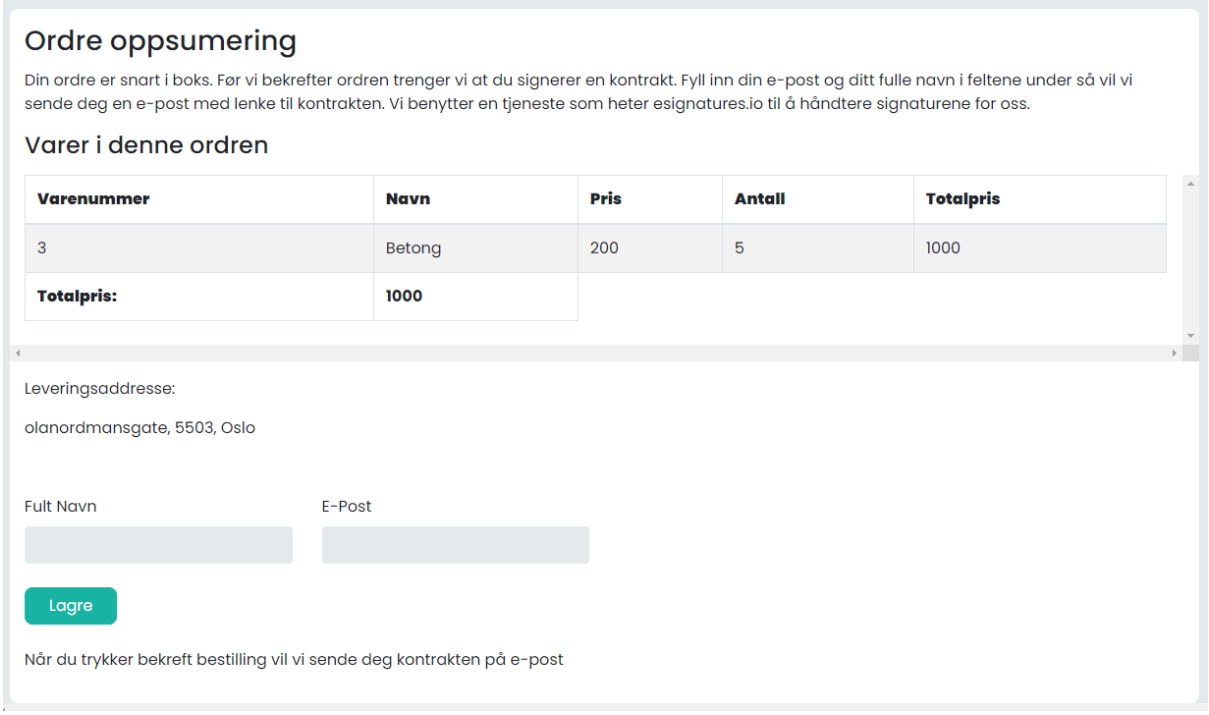
Underveis i brukertestene gjennomførte vi intervjuer med deltagerene for å høre deres tanker og tilbakemeldinger på løsningen. I tillegg til dette observerte vi også noen svakheter i applikasjonen. Vi skal gå over hvilke problemer vi så med applikasjonen og hvordan vi fikset dem.

Figur 29 - Handlekurv før og etter brukertest.

Det første vi observerte da brukerne begynte på bestillingsprosessen var at de fleste ikke forstod at en vare var blitt lagt i handlekurven. Til venstre ser du hvordan handlekurven ser ut uten en indikasjon på hva den inneholder. Til høyre ser du hvordan vi løste problemet; vi la til et ikon som indikerer hvor mange varer som ligger i kurven. Nå har kunden en bekreftelse på at varen ble lagt til, og kan se antall ulike varer som har blitt lagt til i kurven.

Figur 30 - Før og etter bilde av handlekurven.

Gjennom intervjuene fikk vi tilbakemeldinger om at handlekurven var veldig vanskelig å bruke. Vi manglet en funksjon som lot kundene endre antall varer i handlekurven. Vi løste dette ved å legge til et input-felt der det før kun sto antall. Dette gjorde at kundene slapp å gå tilbake til forsiden for å legge til flere varer som allerede lå i kurven.



Figur 31 - Skjermdump av ordreoppsummering.

Gjennom intervjuene fikk vi også vite at flere av kundene syntes ordreprosessen føltes utrygt. De sa at de følte det gikk for fort og savnet et mellomsteg med en ordreoppsummering. Vi la derfor til et ekstra steg hvor man kunne se hele ordren og totalpris før man bekreftet den. Her la vi også inn input-feltene hvor man kan velge hvilken e-post kontrakten skal sendes til.

6.1.3 Vår bruk av Azure DevOps

Da vi tok i bruk Azure DevOps merket vi fort at det ble en bratt læringskurve. Ingen av gruppemedlemmene hadde brukt dette verktøyet før, så det tok litt tid før vi lærte det og kunne bruke det effektivt. Vi hadde også litt problemer med å sette opp pipelines. Pipelines lar deg automatisere hva som skal skje når ny kode blir publisert. Planen vår var å lage en pipeline som kun laster opp koden til produksjonsserver

hvis koden besto den automatiske testen. Dette ble dessverre for tidkrevende å sette seg inn i så vi fant ut at det var bedre å gjøre denne prosessen manuelt. Vi valgte dette fordi vi pleide å pushe til produksjon etter hver Sprint. Det var derfor ikke verdt å bruke mangfoldige timer på å automatisere en prosess som tok 10 - 15 minutter én gang i uka.

Til tross for at vi ikke fikk bruk for pipelines syntes vi Sprintplanlegging gikk veldig fint i Azure DevOps. Det var enkelt å sette opp nye Sprinter og holde styr på backloggen. Vi likte spesielt godt at det kom veldig tydelig frem i grensesnittet hvem som gjorde hva. Vi jobbet på hjemmekontor, og dette sørget for at vi enkelt kunne delegere oppgaver hver dag. Dette fungerte veldig bra med ett unntak: Vi møtte på en bug noen få ganger hvor en oppgave ikke ble oppdatert da et gruppemedlem tildelte seg selv oppgaven. Dette skjedde en gang når vi ikke satt sammen i Discord på grunn av internettproblemer, to medlemmer ble ferdig med oppgaver de hadde tatt på seg i Daily Scrum, og tok derfor på seg en oppgave til fra backloggen. Uheldigvis valgte de samme oppgave, og fordi DevOps var ute av sync ble ikke dette vist på andre medlemmer sitt Dashboard. Det endte opp med at begge medlemmene brukte noen timer på å løse akkurat samme oppgave, som ikke ble oppdaget før branchene skulle merges på slutten av dagen. Dette kostet oss noen tapte arbeidstimer den dagen, men vi forsikret oss mot problemet senere ved at alle arbeidsoppgaver medlemmer jobbet med også var annonsert på Discord i en egen kanal.

6.2 Vurdering av teknisk løsning

I dette kapittelet vurderer vi den tekniske løsningen og hvor fornøyde vi er med resultatet. Vi går over sikkerheten i løsningen og hvilke eventuelle begrensninger den endelige løsningen har.

6.2.1 Kravliste

I lys av kravlisten vi kom fram til i dialog med produkteier i selskapet mener vi at den tekniske løsningen dekker det meste av forventningene gruppen og bedriften hadde

til prosjektet. Vi skal gå gjennom hvilke punkter vi fikk fullført og forklare hvorfor kravspesifikasjonen endret seg underveis.

Funksjonalitet	Implementert	Ikke implementert
Må ha		
Skal utvikles på webplattform og støttes av nettlesere	X	
Admin må kunne invitere kunder til siden for dem	X	
Kunder skal kunne registrere ordre av tilgjengelige varer	X	
Nettsiden må ha et funksjonelt og brukervennlig design	X	
Admin skal kunne opprette bestillinger på vegne av kunder	X	
Løsning må funke på PC, tablet og mobil	X	
Burde ha		
Statistikk som gir verdifull innsikt om kunder på plattformen	X	
Admin skal kunne opprette bestillinger på vegne av kunder	X	
Kontraktsignering for ordre med BankID eller Docusign	X	
Integrere tjenester sånn som XLedger og Viscenario via API		X
Fint å ha		
Løsning kan enkelt videreutvikles for nye bruksområder	X	
Lagerbeholdning kan registreres og endres av admin		X
Servicehistorikk på maskiner i maskinpark		X
Ikke i kravslisten		
Varer er inndelt i varegrupper, og kunder ser begrenset antall varegrupper	X	
Kunder kan endre passord, og admin kan resette passord for kunder	X	
Sikkerhetslogging på alle sensitive endpoints	X	

Tabell 3 - Implementering av kravspesifikasjoner.

Vi har utført all nødvendig funksjonalitet som til sammen skal tilsvare en brukbar e-commerceplattform. Basert på tilbakemeldingene fra brukertestene ser vi også at løsningen fungerer og er forståelig for brukere. Ekstra funksjoner har blitt lagt til i

etterkant av kravlisten fordi nye behov har dukket opp i senere møter med bedriften etter at selve utformingen av løsningen ble tydeligere.

Vår oppdragsgiver leverer varer og tjenester til forskjellige kunder og underveis fant vi ut gjennom dialog med bedriften at det var uoversiktlig å vise alle varer til alle kunder. Det var lite hensiktsmessig å vise kjemikalier for rustbehandling til et byggefirma som kun bestiller glassslipingtjenester. Her trakk vi inspirasjon fra artikkelen vi hadde funnet i forskningsfasen vår hvor det ble understreket viktigheten av å personalisere opplevelsen for kundene (Huang, Chai, Liu og Shen 2019). Vi fant deretter en løsning for å segmentere kundene basert på hvilket behov og ønsker de hadde. Da dukket punktet om ‘varegrupper’ opp i kravlisten. Implementering av denne funksjonaliteten hadde også effekt i å forbedre det mer vage kravet at løsningen skulle kunne videreutvikles til nye bruksområder. Varegruppene vil kunne gjøre at selskapet kan dele opp produkter og tjenester med minimal innsats samtidig som det vil være lettere å legge til ekstra funksjonalitet på enkelte typer varer uten at det vil gjelde for alle varene på plattformen.

Noen av kravene som var ønsket, men ikke påkrevd, fikk vi ikke innvilget grunnet tidsproblemer og grensesnittsrestriksjoner. En integrering med XLedger for å føre inn fakturaer opprettet på plattformen og en med Viscenario for å føre inn nye prosjekter for tjenestebestillinger ville hjulpet med automatisering av arbeidsmengden til bedriften, men det ville vært tidkrevende og ingen av de to tjenestene hadde tilstrekkelig med dokumentasjon på APIene som var enkel å få tak i. Utvikling av disse integrasjonene ville gått utover kjernefunksjonaliteten på plattformen, så vi valgte å se bort ifra disse for dette prosjektet, en vurdering selskapet også var enig i. Ønskene om et lagersystem og servicehistorikk var vanskelig å få logisk presentert i løsningen vi hadde. Skulle vi koblet opp lagersystem med ordre slik at det kom fram om det var mangler av produkter bestilt ville dette være litt mer komplisert på backenden vår, så den implementeringen ville vi mest sannsynlig ikke hatt tid til. I tillegg hadde selskapet selv dårlige prosesser for å håndtere lagerbeholdning i dag, som gjorde at det ville vært vanskelig for bedriften å ta i bruk uten ekstra opplæring og utarbeiding av rigide prosesser. Av disse grunnene så vi oss nødt til å nedprioritere både lagersystemet og servicehistorikken.

Løsningen er forbi prototypefasen og er klar for at selskapet tar den i bruk i noen av sine bestillinger allerede etter endt bachelorprosjekt. Løsningen er skalerbar og kan bygges på etter behov. Vi tror at andre utviklere lett kan sette seg inn i kodebasen og fortsette utviklingen siden vi har dokumentert APIet og koden grundig.

6.2.2 Sikkerhet

Vi er også fornøyd med sikkerheten i applikasjonen; alle endpoints er sikret og vi har også sørget for at kritiske sikkerhetsadvarsler kommer tydelig frem i frontenden om de skulle oppstå. Disse advarslene er ment for å varsle admins om mistenkelig aktivitet. For å vise dette med et eksempel: En innlogget kunde ved navn Ola Bygg A/S prøver å hente ut en ordre tilhørende en annen kunde ved navn Kari Entreprenør A/S. Serveren vil se at en innlogget bruker ved navn Ola Bygg A/S prøver å hente ut informasjon vedkommende ikke har tilgang til. Serveren vil derfor loggføre dette som mistenkelig.

```
res.status(ECode.FORBIDDEN).send();
Logger.log('crit',`Potensiell Kritisk sikkerhets varsel:
bruker med id: ${req.user.id} og brukernavn: ${req.user.username}
prøvde å hente ut en ordre tilhørende bruker med id: ${req.params.id},
dette ble stoppet av serveren. Om dette var en feil så prøv på nytt
med en bruker som har tilgang til denne handlingen.`)
```

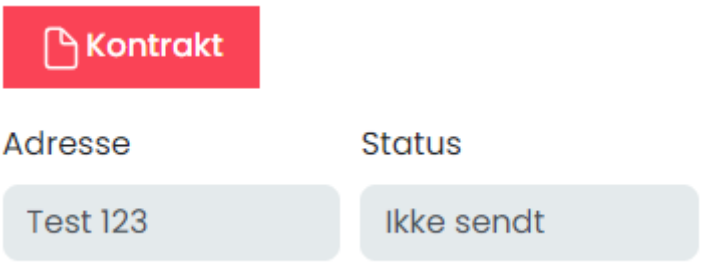
Figur 32 - Eksempel fra koden på sikkerhetslogging

Logikken bak dette er at denne handlingen ikke er støttet gjennom brukergrensesnittet. Det vil si at for å sende denne forespørselen til serveren må Ola Bygg A/S lage denne forespørselen manuelt. Vi antar at en vanlig kunde hverken har kunnskap eller motiv for å gjøre dette og velger derfor å logge det som vist på bildet over.

6.2.3 Designprinsipper

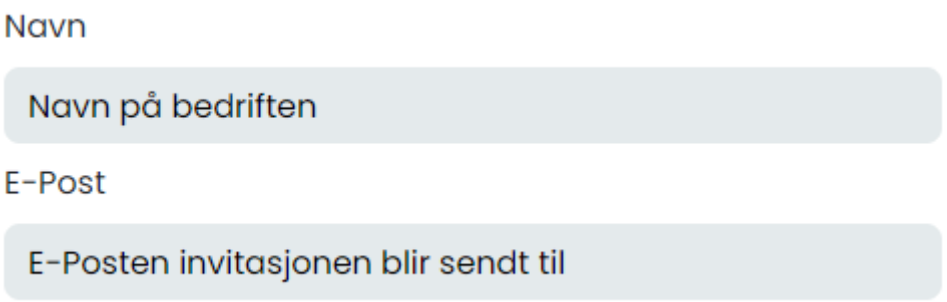
Løsningen tar i bruk et enkelt, men funksjonelt grensesnitt. Her har vi som nevnt tidligere brukt prinsippene i boken “The Design of Everyday Things” av Don Norman.

Målgruppen vår består av mennesker som ikke innehar en høy IT-kunnskap og det er derfor viktig å ha en universell utforming som allmennheten kan sette seg kjapt inn i og ta i bruk.



Figur 33 skjermdump av ordreinformasjon

På bildet over ser du et utklipp fra ordresiden. På kontraktknappen tar vi i bruk en rød farge. Dette passer ikke veldig bra inn med resten av fargepaletten, men til gjengjeld blir rødt ofte forbundet med ikonet til en PDF-fil. Med andre ord har vi **ekstern konsistens** med det universelle symbolet for PDF. Det er også i tråd med prinsippet om **synlighet** siden folk vil anta at en knapp som representerer PDF vil åpne en PDF-fil. Teksten i input-feltene er også grå istedenfor svart for å gjøre det mer **synlig** for brukeren at det ikke er redigerbart.



Figur 34 - Skjermdump av input-felter.

Input-feltene våre tar i bruk det samme designet gjennom hele applikasjonen og spiller på **sammenheng**. Hvert felt har tilhørende tekst som forklarer hva slags informasjon som skal tastes inn i hvert felt. Prøver man å sende inn informasjon uten å fylle ut alle eller om noen av feltene inneholder ugyldig informasjon vil brukeren få **tilbakemelding** på dette.

Vi har brukt mange gjenbrukbare komponenter i designet vårt. Tabellene som brukes i administratorportalen er helt like. Det eneste som skiller dem er informasjonen de inneholder slik at søk og sortering foregår på samme måte. Dette er i tråd med prinsippet om **intern konsistens**.

Figur 35 - Skjermdump av vellykket og feilet toasts.

I tillegg har vi en toast-funksjonalitet som gir **tilbakemelding** når brukeren har gjort noe, med et **konsistent** design over hele nettsiden. De er alltid posisjonert i bunnen av siden og tar i bruk logoer og farger for å raskt kommunisere til brukeren om handlingen ble utført eller feilet.

Figur 36 - Skjermdump av handlekurv-knappen på sidemeny.

Handlekurven på kundeportalen tar i bruk **hint.** En av observasjonene vi gjorde under brukertesten var at mange slet med å forstå at en vare hadde havnet i handlekurven. Dette var fordi det ikke var noe hint eller tilbakemeldinger hvis man la noe i kurven. Derfor endret vi designet slik at man får opp en tilbakemelding og et hint når man legger en vare i handlekurven.

Vi har også lagt inn **begrensninger** i løsningen for å sørge for at brukere ikke kan gjøre feil. Et eksempel på dette er varegruppesystemet. Oppdragsgiver har forskjellige produktutvalg for forskjellige kunder og ikke alle kundene har tilgang på de samme produktene. Vi har derfor sørget for at kun produkter som kan bestilles er tilgjengelig i grensesnittet.

6.2.4 Begrensninger

Det er dessverre noen mangler som vi skulle likt å adressere hadde vi hatt tid. Først og fremst så mangler vi automatiske tester for apiet. Frontenden mangler også unit tester på komponentene. Dette kan føre til uforutsette bugs når deler av kodebasen endres og svakheter kan dermed oppstå som et resultat av at bugs “går under radaren”.

En annen mangel er dokumentasjon for brukere. Vi har dokumentert prosjektet grundig for utviklere, men det mangler dokumentasjon som forteller hvordan vanlige brukere kan ta i bruk systemet. En løsning for fremtidig utvikling kan være å lage en interaktiv læringsplattform integrert i løsningen som kan lære opp nye brukere til å bruke løsningen.

6.3 Vurdering av nytte- og forretningsverdi

Da vi skulle prioritere de forskjellige funksjonene på kravspesifikasjonen vurderte vi dem etter hvor stor verdi de ville ha for oppdragsgiver. Denne tankegangen holdt vi oss til gjennom hele prosjektet, også da det dukket opp nye interessante problemstillinger underveis i prosessen.

En av disse var oppdragsgiver sitt ønske om å segmentere kunder, spesielt når de skulle bryte inn i nye markeder. Her var problemet at alle varer og tjenester var synlig på plattformen for alle kunder, så vi løste dette ved å legge til den tidligere nevnte varegruppeinndelingen. Denne nye løsningen gjorde også at oppdragsgiver manuelt kunne opprette en “midlertidig” tjeneste som kun var tilknyttet en enkelt ordre, og derfor ikke en del av det ordinære vareutvalget. Dette var spesielt nyttig for oppdragsgiver siden de nå kunne legge inn avtaler de hadde gjort via e-post om tjenester som ikke lå i utvalget fra før, eller om de hadde blitt enige om å gå etter andre priser.

Som diskutert tidligere gjorde vi brukertester for å sammenligne løsningen vår mot e-postprosessen til selskapet i dag. Hensikten var å vurdere om vår plattform kunne tas i bruk av selskapets kunder uten å gjøre brukeropplevelsen enda verre. Det vi så

var at løsningen vår fungerte bedre i den begrensede testrunden vi hadde basert på resultat av SUS-rangering, gjennomføringstiden og intervjuene etter brukertestene. Disse resultatene og forskningen vi fant frem ble også drøftet med bedriften, som også mente dette var et godt tegn på at løsningen vår ville være fordelaktig, eller i verste fall ikke føre til store problemer med overgangen, også for deres kunder. På grunnlag av dette har vi konkludert med at både oppdragsgiver og vår gruppe er trygge med å lansere produktet ut mot kunder uten at det vil skade kundeforhold, som var et av de viktigste kravene til Watec fra dag én.

En annen viktig funksjon vi fikk implementert var mulighet for administratorer til å opprette ordre på vegne av kunder. Oppdragsgiver antok at det ville ta litt tid å implementere hele systemet inn i prosessen sin og få alle kundene inn på plattformen. Derfor la vi til rette for at bedriften kan fortsette å bruke mail på noen av kundene om ønskelig. Om de mottar en bestilling over mail kan de enkelt legge denne inn i systemet slik at de slipper å be kunden logge inn og gjøre det selv, og bedriften vil fortsatt få nytte ved at bestillingene er samlet på samme plattform.

6.4 Videreutvikling

Ved videre utvikling av prosjektet bør man først og fremst fokusere på å få på plass god testdekning på grensesnittet. Deretter vil det være integrasjoner med andre plattformer som bør prioriteres. Oppdragsgiver benytter Xledger, som er et regnskapsføringsprogram som tilbyr et API. Fullførte ordre bør automatisk sendes til Xledger slik at faktura kan sendes ut.

En annen tjeneste som bør integreres er Viscenario, som er prosjektstyringsprogrammet selskapet bruker når de skal levere tjenester til kunder. Signerte bestillinger av tjenester bør opprette nye prosjekter i Viscenario automatisk. En annen funksjon som kan være nyttig er et lagersystem. Per dags dato har ikke løsningen vår mulighet til å følge med på lagerstatus på de ulike produktene. Det hadde vært en fordel med et slikt system siden oppdragsgiver benytter seg av underleverandører for noen av produktene. Det kan ta lang tid å få tak i nye produkter fra disse leverandørene, så et system som varsler når lageret holder på å

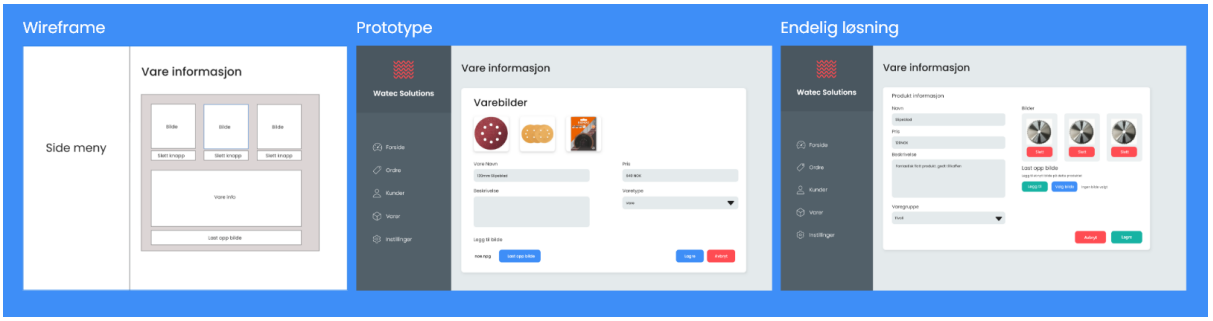
gå tomt hadde vært en fordel. Det ville også åpnet for muligheten å kommunisere lagerstatus til kunder og følge bestillinger som blir gjort fra start til slutt.

Det ville også vært fordelaktig å implementere et API som ville stått åpent for kundene å ta i bruk. Dette ville latt kunder integrere innkjøpsprosessen i sine egne systemer. For å implementere noe slikt vil det være en del ekstra arbeid som må gjøres, spesielt rundt sikkerhetsaspektet for å forsikre at kunder ikke kan få tilgang på intern data. Vi har derimot et API som er dokumentert og sikret, samtidig som det aktivt sikkerhetslogger forsøk på innhenting eller sending av data bruker ikke skal ha tilgang på. Vi ser for oss at APIet vi har bygget kan bli gjort mer åpent i fremtiden hvis man implementerer for eksempel JSON Web Tokens for å sikre endpoints i stedet for å gjøre alt session-basert gjennom Passport slik det er i dag.

7 Konklusjon

Hvordan kan vi utvikle en smartere løsning for å håndtere ordre som sparer bedriften for tid og penger, uten å gi negativt utslag på de gode kundeforholdene de har i dag?

For å besvare denne problemstillingen har vi brukt mye tid på å sette oss inn i hvordan oppdragsgiver gjennomfører en salgsprosess i dag. Vi begynte først å kartlegge hvilke behov bedriften hadde, for å så oversette behovene til tekniske funksjoner som kunne innfri disse behovene. Dette ble utgangspunktet fra vår kravspesifikasjon. Vi benyttet Scrum som prosjektmetodikk, så en del av prosessen gikk ut på å iterere over denne listen for å se hva som fungerte og hva som måtte endres. Vi hadde flere møter med bedriften for å få tilbakemeldinger på admin-delen av løsningen, og vi gjennomførte brukertester for å teste kundedelen.



Figur 37 - Skjermdump av utviklingsprosess.

Arbeidsprosessen vår har gått ut på å gjennomføre to eller tre Sprinter med noen definerte mål som skulle tilføre verdi til løsningen. Etter Sprintene var ferdig ble de vurdert, og hvis oppgavene i Sprinten ble definert som ferdige ble de tatt med i neste møte med bedriften. Der fikk vi tilbakemeldinger på de siste Sprintene. Disse tilbakemeldingene ble tatt i betraktning da vi skulle planlegge de neste Sprintene. Den hyppige kontakten mellom oss og bedriften har gitt oss mye verdifull innsikt og tilbakemeldinger underveis i utviklingen. Vi tror derfor funksjonene som ble tatt med i den endelige løsningen tilfører god nytteverdi for bedriften.

Basert på brukertestene vi gjennomførte mot løsningen har vi også fått gode tilbakemeldinger på grensesnittet og brukervennligheten av plattformen, spesielt sett opp imot en bestillingsprosess som skjer bare over mail. I tillegg til dette er løsningen

bevisst laget for å være fleksibel for bestillinger som går utover det bedriften har lagt ut på plattformen eller der kunden selv ønsker å gjøre bestillinger over e-post, da admin kan legge inn bestillinger på vegne av kunder med tilpassede tjenester. I lys av dette ser vi ingen grunn til at tjenesten vil komme på bekostning av kundeforhold, da grensesnittet kan prestere bedre enn å gjøre det via e-post samtidig som bestillinger over e-post fremdeles er støttet.

Oppdragsgiver syntes produktet var godt strukturert med godt design siden det var veldig lett for dem å finne informasjonen de lette etter. De var også veldig fornøyd med brukervennligheten på grensesnittet, og de mente at det ville være lett for nye kunder å ta i bruk tjenesten. De var også positive til statistikken på hjemmesiden, og at nye bestillinger kom opp under ‘siste varsler’ her.

Et viktig element som kom med i slutten av prosjektet som var spesielt relevant for selskapet var varegruppefunksjonaliteten. Det at ansatte selv kan legge inn nye varer og tildele dem varegrupper gjør at plattformen er mer skalerbar enn det som først var planlagt. Dette vil gi selskapet mulighet til å skille ut sine ulike kunder etter hvilke behov de har og utvide sin porteføljе av varer og tjenester inn i nye bransjer uten å måtte budsjettere videreutvikling av bestillingsplattformen.

8 Referanser

Berthon, Pierre, Leyland Pitt, Jean-Paul Berthon, Colin Campbell, og Des Thwaites. 2008. "e-Relationships for e-Readiness: Culture and corruption in international e-B2B" *Industrial Marketing Management,* 37(1):83-91 https://doi.org/10.1016/j.indmarman.2007.06.014

Clark, Lin. 2018. *A crash course in just-in-time (JIT) compilers.* Oppdatert 28 Februar, 2017. https://hacks.mozilla.org/2017/02/a-crash-course-in-just-in-time-jit-compilers/.

Fauska, Polina, Natalia Kryvinska, og Christine Strauss. 2013. "E-commerce and B2B Services Enterprises", *27th International Conference on Advanced Information Networking and Applications Workshops*, 1141-1146 https://doi.org/10.1109/WAINA.2013.98

Fielding, Roy Thomas. 2000. "Architectural Styles and the Design of Network-based Software Architectures". Doktoravhandling, University of California

Huang, Yadong, Yueting Chai, Yi Liu og Jianping Shen. 2019. "Architecture of next-generation e-commerce platform". *Tsinghua Science and Technology,* 24(1):18-29. https://doi.org/10.26599/TST.2018.9010067

McLean Graeme J., 2017, "Investigating the online customer experience – a B2B perspective", *Marketing Intelligence & Planning*, 35(5):657-672. https://doi.org/10.1108/MIP-12-2016-0222

Murphy, Kenneth E. og Steven John Simon. 2001. "Using cost benefit analysis for enterprise resource planning project evaluation: a case for including intangibles." *Proceedings of the 34th Annual Hawaii International Conference on Systems Sciences,* 11-?. https://doi.org/10.1109/HICSS.2001.927131

Urlikas, Tomas. 2020. "Figma vs. Adobe XD: How we picked our next design tool". Oppdatert 12 Juni, 2020. https://uxdesign.cc/figma-vs-adobe-xd-how-we-picked-our-next-design-tool-e8a8f5d24cca

Nielsen, Jakob. 2000. "Why You Only Need to Test with 5 Users". Oppdatert 18. Mars 2000. https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users

Norman, Don. 2013. *The Design of Everyday Things: Revised and Expanded Edition*. 1. utg. New York: Basic Books

Rubin, Kenneth S. 2001. *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. With foreword by Mike Cohen. 1. utg. Ann Arbor: Addison-Wesley

Sauro, Jeff. 2011. "SUStisfied? Little-Known System Usability Scale Facts". Oppdatert 11. August 2011. http://uxpamagazine.org/sustified/

SimilarTech. 2021. “NodeJs Market Share and Web Usage Statistics”. Oppdatert 13. Mai 2021. https://www.similartech.com/technologies/nodejs

Sutherland, Jeff, Anton Viktorov, Jack Blount, og Nikolai Puntikov. 2007. “Distributed Scrum: Agile Project Management with Outsourced Development”, *40th Annual Hawaii International Conference on System Sciences (HICSS’07)* https://doi.org/10.1109/HICSS.2007.180

Toftøy-Andersen, Eli og Jon Gunnar Wold. 2012. *Praktisk Brukertesting*. 1. utg. Oslo: Cappelen Damm

Węglarz, Rafał. 2020. “9 Companies that Use React”. Oppdatert 20. November, 2020 https://www.droptica.com/blog/9-companies-use-react/

World Wide Web Consortium. 2018. “Web Content Accessibility Guidelines (WCAG) 2.1”. Oppdatert 5. Juni, 2018. https://www.w3.org/TR/WCAG21/#requirements-for-wcag-2-1

World Wide Web Consortium. 2021. “Understanding Success Criterion 1.4.3: Contrast (Minimum)”. Oppdatert 9. Mars, 2021. https://www.w3.org/WAI/WCAG21/Understanding/contrast-minimum.html

Yang, Ying, Paul Humphreys, Ronan McIvor. 2003. “The impact of E-Commerce on B2B Services”. *The 8th Cambridge Symposium on International Manufacturing (CamSIM)*, pp 314-322. https://www.researchgate.net/publication/242450666

9 Vedlegg

Vedlegg A: Prosjektbeskrivelse

Watec Solutions er et selskap i Lillestrøm opprettet i 2016 som leverer ulike tjenester og produkter for bedrifter. Tjenestene og produktene har hensikt å være kostbesparende for selskaper og miljøvennlig for samfunnet. Watec prioriterer gode kundeforhold og kundeopplevelser, selskapet tjener best på faste kunder og volum. I de siste årene har mesteparten av arbeidet til Watec gått ut på glassliping tjenester i anleggsbransjen. Dette går ut på å slipe vekk skraper og skader i glass som forekommer under utbygging i stedet for å bytte ut hele glasset, som er både billigere for entreprenøren samt et mer miljøvennlig alternativ. I tillegg til dette driver selskapet med forskning og utvikling av andre løsninger, og har ambisjoner om å jobbe inn mot flere bransjer. Blant annet jobber selskapet med å utvikle kjemi for å forhindre korrosjon av metaller, samt produksjon av maskiner som kan gjøre plastikk tilbake om til olje.

Problemstilling

Watec Solutions mangler idag et system for å samle ordre, lagerbeholdning og oppfølging av maskinparken. De ønsker et oversiktlig system som gir en oversikt over hvem kunder som har bestilt hva og hvilke produkter de har på lager. De ønsker også at dette skal integreres med ulike programmer de tar til bruk i dag, eksempelvis Xledger og Viscenario, slik at systemet kan hente inn data selskapet produserer på andre plattformer. De vil også at systemet kan holde styr på service historikken til maskinparken deres.

Teknologisk Løsning

Løsningen må innfri forventningene som er beskrevet over. Vi ser for oss en løsning der vi deler opp funksjonaliteten i 2 deler. Kundens perspektiv og oppdragsgivers perspektiv.

Kunden

Kunden trenger å kunne opprette en bruker. Dette har vi sett for oss kan gjøres gjennom en funksjon som lar oppdragsgiver sende en mail med en registreringslink. Deretter vil kunden få tilgang til sider som lar dem opprette ordre og kommunisere med bedriften. Kunden skal også kunne legge inn kommentarer på ordre og signere med BankID eller Docusign. Vi ser også for oss en løsning der det finnes flere "kundegrupper". Dette vil gjøre slik at bedriften kan vise visse produkter kun for en bestemt kundegruppe.

Oppdragsgiver

Fra oppdragsgivers perspektiv så trenger de flere systemer til å samle driften i et program. Vi har valgt å dele programmet inn i 4 deler.

Lagerstyring
Bedriften skal kunne legge inn varer i systemet slik at de kan holde styr på hva som er på lager. Vi ser for oss at dette kan gjøres ved at kunden kan legge inn varer i et register. Når disse varene skal legges inn i systemet kan de bare taste inn ett varenummer og antall også blir det lagt inn i lagerbeholdningen. Dette kan også forenkles ved å f.eks legge til en funksjon der man også kan registrere strekkoder på varer og på den måten bare scanne inn varene.

Ordresystem
Ordre systemet skal håndtere kunde ordre og kommunisere med Lageret slik at lagerbeholdningen blir oppdatert når kunder legger inn bestillinger. Her skal bedriften kunne se nye ordre og sjekke status på eldre ordre.

Servicehistorikk
Oppdragsgiver har en maskinpark som regelmessig må på service. Derfor ønsker vi å legge til en funksjon der hver maskin har en id. Når denne maskinen har vært på service kan oppdragsgiver plotte dette inn i systemet. Om 6 måneder når maskinen må på service igjen så vil systemet sende et varsel om at maskinen må på service igjen.

Kunderegister

Tjenesten trenger et kunderegister der oppdragsgiver kan legge til nye kunder og se informasjon om allerede eksisterende kunder. Her skal informasjon om bedriftens kontaktperson være tilgjengelig, samt kundens ordrehistorikk og kundegruppe.

Målsetninger

Vi ønsker å levere et IT system av høy kvalitet som løser bedriftens problemer med en uoversiktlig verdikjede. Vår mål er å levere et produkt som innfrir de tidligere nevnte kravene.

Knytning til forskningsområde

For å utarbeide en god løsning må vi tilegne oss kunnskap om flere ulike teknologier. Siden selskapet ikke har noen IT avdeling er det også viktig at vi lager et skalerbart system som ikke trenger så mye vedlikehold. Vi tenker derfor å se nærmere på skylagring. Vi skal også lese oss opp på litteratur som omhandler logistikk systemer slik at vi kan lage en effektiv og intuitiv løsning.

Ambisjonsnivå

Ambisjonen vår er å levere en nyttig løsning for Watec Solutions som bidrar til å forbedre prosesser i selskapet, samt at løsningen blir utviklet på en god måte slik at vi kan skrive en god bacheloroppgave.
Vi sikter mot en best mulig karakter, og vil streve for dette. Vi vet at vi har kompetansen til å utvikle noe godt og vil derfor ha A som vårt karaktermål, med forbehold om at vi vil akseptere en B uten å klage.

Behov for spesialkompetanse

Vår løsning kan kreve at vi innhenter noen kompetanse innen ERP-systemer, som vi ikke har mye erfaring med selv. Vår interne veileder har god erfaring i dette området og kan gi oss råd om hvor denne informasjonen og kunnskapen kan innhentes, så det kan kreve at vi

bruker dette som en ressurs underveis. Hovedmålet vårt er at vi lærer all informasjonen som er nødvendig for å produsere løsningen, og heller lener oss på individer med spesialkompetanse for å finne ut av hvordan vi best tilegner oss all kunnskap vi trenger. Vi ønsker ikke å være avhengige av noen utenfor gruppen for at sluttproduktet blir ferdigstilt, all kompetansen bør ligge hos gruppen, og spesialkompetanse utenfor gruppen skal hentes inn for å lære dette.

Vedlegg B: Risikoplan

	Risiko	Konsekvens (0 - 1)	Sannsynlighet (0-10)	Risiko (0 - 10)	Tiltak for å unngå risiko	Tiltak hvis problemet oppstår
1	Feilberegning av scope	0,9	4	3,6	Gå gjennom oppgaven sammen med ekstern og intern veileder.	Restrukturere kravene til oppgaven i samarbeid med ekstern og intern veileder
2	Manglende kompetanse	0,3	5	1,5	Holde oss oppdatert på rammeverk og kodespråk vi velger å ta i bruk.	Tilegne oss ny kunnskap eller se etter ekstern hjelp
3	Sykefravær	0,6	6	3,6	Følge gjeldende råd fra FHI ang. korona pandemien.	Reorganisering av tildelte oppgaver
4	Endring i kravspesifikasjon	0,5	5	2,5	God planlegging og flytende dialog med bedrift og veileder	Legge endringer inn i product backlog og få inn på en senere Sprint.
5	Dårlig kommunikasjon innad i gruppen	0,3	4	1,2	Ta i bruk prosjektstyring program og	Oppdatere prosjektstyring program og ta en gjennomgang av

					opprette rutiner på hvilke kanaler vi bruker til å kommunisere.	rutinene
6	Mangel på arbeidssted grunnet COVID-19 restriksjoner	0,6	8	4,8	Legge opp gode løsninger for hjemmekontor	Hjemmekontor og digitale møter over Zoom eller Discord
7	Konflikter innad i gruppen	0,4	3	1,2	Holde en respektfull tone og lytte til hverandre.	Løse problemene sammen gjennom en dialog. Eventuelt ta kontakt med veileder
8	Tap av arbeid	1	0,35	0,3	Ta i bruk Git og sørge for å pushe til Git ofte	Gå sammen om å erstatte det tapte arbeidet
9	Internettproblemer	0,5	6	3	Holde lokal versjon av kode oppdatert	Ta i bruk 4G fra mobilen eller jobbe offline og kommunisere gjennom sms
10	Ta i bruk et bibliotek som inneholder sikkerhetshull	1	7	7	Alltid gjøre grundig research før man tar i bruk nytt bibliotek	Erstatt biblioteket eller sett i gang tiltak for å tette sikkerhetshullet
11	Manglende dokumentasjon på selvskreven kode	1	8	8	Skrive gode, informative kommentarer ved all kode som trenger forklaring	Finne ut hvem som har skrevet koden og få vedkommende til å gå over og dokumentere
12	Feil bruk av scrum /	1	7	7	Alltid holde seg til gjeldende Sprint	Scrum Master inviterer til et felles møte hvor vi går

	manglende oppfølging av Sprint backlog eller produkt backlog				backlog og oppdatere underveis	gjennom Sprint eller produkt backlog og retter eller oppdater til vi er i rute.
13	Bedriften blir ikke fornøyd med resultatet	1	3	3	Holde en god dialog med bedriften gjennom hele prosessen. Ta deres bekymringer og tilbakemeldinger på alvor	Ta et møte med bedrift der vi utreder hva som kan gjøres. Sette i gang tiltak for å forbedre produktet slik at bedriften blir fornøyd

Vedlegg C: Samtykkeerklæring

Are you interested in taking part in the research project

”En Analyse av Effekten Valg av Plattform har på Kundeopplevelse i Digitale Salgsprosesser”?

This is an inquiry about participation in a research project where the main purpose is to study the effect on customer experience on different digital sales platforms. In this letter we will give you information about the purpose of the project and what your participation will involve.

Purpose of the project

The purpose of this project is to study the effects on customer experience on different sales platforms. We are going to do user testing on 2 different platforms to see if the user experience is different, better or worse. We also want to understand why it's different and what key factors play a role in the users perception of a platform. We will also do interviews on the participants as well as ask them to fill out a survey at the end of the test. This project is tied to our exam in the course “Undersøkelsesmetoder” at Høyskolen Kristiania. The results of the data we collect will also be used in our bachelor’s thesis but will be stripped of personal information regarding the participants. The only personal information that will be used is: Age, Position / Career, level of IT knowledge and previous experience with sales platforms.

Who is responsible for the research project?

Kristiania University College is the institution responsible for the project.

Why are you being asked to participate?

You have been asked to take part in this study because you meet our demographic criteria either because of your age or your career.

What does participation involve for you?

If you choose to take part in the project, this will involve that you participate in a user test of a digital plattform, recorded electronically over Zoom. Estimated time of the test is 15 minutes, but expect that it could take up to 30 minutes in case of any technical issues using Zoom. After the session you will be asked to fill out an online survey about the test you just participated, which should take approx. 5 minutes.

Participation is voluntary

Participation in the project is voluntary. If you choose to participate, you can withdraw your consent at any time without giving a reason. All information about you will then be made anonymous. There will be no negative consequences for you if you choose not to participate or later decide to withdraw.

Your personal privacy – how we will store and use your personal data

We will only use your personal data for the purpose(s) specified in this information letter. We will process your personal data confidentially and in accordance with data protection legislation (the General Data Protection Regulation and Personal Data Act).

- The personal data about you will be available to the group performing the study. All members of the group will have access to the zoom recordings as well as responses from the interviews and survey. The group members are:
    - Jonas Kalmar Rønning
    - Christian Iversen
    - Kristoffer Kittilsen.
- The raw video footage will be kept safe on an encrypted hard drive and your answers will be stripped of personal information such as your name in the final paper.
- The team member responsible for storing the video footage is Jonas Kalmar Rønning.

You will not be recognizable in the final paper and the only information about you that will be displayed are:

- Age
- Career
- Level of IT knowledge
- Previous experience with B2B sales platforms.

What will happen to your personal data at the end of the research project? The project is scheduled to end 13. April, 2021. At the end of this date, all personal information stored about you, including your name, as well as the digital recording of your session will be deleted. Only the anonymized data used in the report will remain.

Your rights

So long as you can be identified in the collected data, you have the right to:

- access the personal data that is being processed about you
- request that your personal data is deleted
- request that incorrect personal data about you is corrected/rectified
- receive a copy of your personal data (data portability), and
- send a complaint to the Data Protection Officer or The Norwegian Data Protection Authority regarding the processing of your personal data

What gives us the right to process your personal data?

We will process your personal data based on your consent.

Personal data will be processed in accordance with the guidelines of Kristiania University College, and in accordance with data protection legislation in Norway.

Where can I find out more?

If you have questions about the project, or want to exercise your rights, contact: ● Høyskolen Kristiania via Kjeld Hansen, by email: Kjeld.Hansen@kristiania.no. ● Project Leader Christian Nicolai Iversen, by email: christian.nicolai.iversen@gmail.com or by telephone: +47 928 20 670

● NSD – The Norwegian Centre for Research Data AS, by email: (personverntjenester@nsd.no) or by telephone: +47 55 58 21 17.

Yours sincerely,

Christian Nicolai Iversen

---------- Consent form

I have received and understood information about the project “En Analyse av Effekten Valg av Plattform har på Kundeopplevelse i Digitale Salgsprosesser” and have been given the opportunity to ask questions. I give consent:

- to participate in recorded user test and interview over Zoom
- to participate in filling out a questionnaire after the test

I give consent for my personal data to be processed until the end date of the project, approx. 13. April, 2021

---------- (Signed by participant, date)

Vedlegg D Begreper

Begrep	Forklaring
API	API står for Application Programming Interface og er et grensesnitt som spesifiserer hvordan 2 applikasjoner kan kommunisere med hverandre
Azure Pipelines	Azure Pipelines lar deg sette opp automatiske handlinger som blir utført når ny kode blir pushet til Gitmappen.
B2B	B2B står for business to business. Et begrep som brukes om bedrifter som selger produkter og tjenester til andre bedrifter.
Backend	Backend referer til delen av en applikasjon som kunn utviklerne ser. Det er her database funksjoner og business logikk blir kjørt.
Bootstrap	Bootstrap er et CSS-bibliotek som inneholder ferdig designede html komponenter
Branch	En branch eller på norsk "gren" beskriver en del av en Git mappe. En git mappe kan inneholde mange grener som har forskjellige versjoner av koden.
Brukertesting	Brukertesting er et begrep som brukes for å beskrive prosessen der man lar brukere teste et produkt eller applikasjon med hensikt å vurdere hvordan det fungerer i praksis.
Bug(s)	En bug er en feil i et datasystem.
Burndowngraf	En burndowngraf er en graf som viser hvor mye arbeid som er gjort i et spesifisert tidsrom i prosjektmetodikken scrum
C#	C# (utales c-sharp) er et programmeringspråk.
CRM	Customer Relationship Management. Et begrep som beskriver en type programmvare som holder styr på kunde relasjoner.
DocuSign	En tjeneste som lar deg signere dokumenter digitalt.
E-commerce	Det engelske ordet for e-handel. Beskriver en handel som er utført over Internett

ERP	Enterprise Resource Planning. Beskriver en programmvare som brukes til å styre prosessene i en bedrift som for eksempel lagerbeholdning og lønningslister
Express	Express er et bibliotek som lar deg lage en webserver i JavaScript
Frontend	Frontend referer til delen av en applikasjon som kjører på klienten til sluttbrukeren. Alt brukeren kan se og interagere med er en del av frontenden
GET	GET er en type forespørsel der man ber om å få en ressurs gjennom HTTP protokollen
Git	Git er et versjonskontroll system som lar deg spore endringer i koden over tid. Det fungerer ved å la deg laste opp forskjellige versjoner av koden. Flere brukere kan laste opp nye versjoner som så blir "smeltet" sammen til en felles kodebase
HTTP	Hyper Text Transfer Protocol er en protokoll som definerer hvordan man applikasjoner skal kommunisere. Det er denne protokollen som brukes på World Wide Web (WWW).
JSON	JavaScript Object Notation er en standard som spesifierer hvordan man skal formulere tekst når man kommuniserer med en annen applikasjon. Den er basert på objekt strukturen til kodepsråket JavaScript
JSON Web Token	JSON Web Token er standard som spesifiserer hvordan man kan autentisere en bruker eller en applikasjon ved å gi dem en token for å bekrefte identiteten
Merge	Merge er en handling som utføres i versjonskontrollsystemet Git. Når du merger to filer så kombinerer du dem til en fil.
Node.js	Node.js er et rammeverk som lar deg kjøre JavaScript kode utenfor nettleseren.
NoSql	NoSql beskriver en database som ikke benytter seg av SQL strukturen. Istedenfor å lagre data som tabeller vil en NoSql database lagre dokumeter i JSON format
Passport	Passport er et JavaScript bibliotek som håndterer autentisering.
PostgreSQL	PostgreSQL er en relasjonsdatabase som implementerer SQL.

Produktbacklog / Product Backlog	Produktbacklog er en liste over arbeidsoppgaver som må fullføres for at produktet skal bli ferdig.
Produkteier	Begrepet produkt eier referer til en rolle i scrum metodikken. Produkt eier er personen som er ansvarlig for produktet.
Relasjonsdatabase	En relasjons database er en database hvor dataen i en tabell kan være avhengig eller ha en tilknytning til data som ligger i en annen tabell
REST	Representational State Transfer er en arkitekturstil som spesiferer hvordan to applikasjoner kan kommunisere.
Scope	Scope beskriver en menge med arbeid eller et sett med funksjoner en applikasjon skal ha.
Scope Creep	Scrope Creep er når man legger til flere funksjoner til et Scope allerede under utviklingen, slik at arbeidsmengden øker. Scope Creep benyttes vanligvis til å beskrive når man legger til for mange funksjoner slik at man risikerer å ikke få ferdigstilt applikasjonen
Sprint	En sprint beskriver et tidsrom i prosjektmetodikken scrum. En sprint på 5 dager vil bety at det er 5 arbeidsdager i den sprinten.
Sprint Review	Sprint review er en del av scrum metodikken og går ut på at man skal vurdere hva man har laget i en sprint. Man skal med andre ord vurdere resultatet av sprinten
SQL	Structured Query Language er et språk man bruker for å kommunisere med en database.
SUS	System Usability Scale er en skala man bruker for å rangere brukervennelighet. Man stiller brukere spørsmål der de kan gi svar fra 1 til 5. Når man legger sammen svarene får man en rangering man kan bruke for å se hvor brukervennelig en applikasjon er
WCAG	Web Content Accessibility Guidelines er et sett med retningslinjer man bør følge for å sørge for at en applikasjon er universelt utformet.